

ООО «Базальт СПО»
Институт Программных Систем РАН

**Двадцатая конференция
Свободное программное обеспечение
в высшей школе**

Переславль-Залесский, 07–09 февраля 2025 года

Сборник материалов конференции
Научное издание

Москва,
МАКС Пресс,
2025

УДК 004.91:378
ББК 32.97:74.48
Д22



<https://elibrary.ru/njbljm>

Программный комитет:
А. А. Савченко, — председатель,
А. В. Бондарев, Г. В. Курячий, А. А. Маркина, М. О. Петрова

Д22 **Двадцатая конференция «Свободное программное обеспечение в высшей школе»** : материалы конференции / Переславль-Залесский, 07–09 февраля 2025 г. / отв. ред. Чёрный В. Л. — М.: МАКС Пресс, 2025. — 144 с.
ISBN 978-5-317-07341-1
DOI: <https://doi.org/10.29003/m4350.978-5-317-07341-1>
EDN: <https://elibrary.ru/njbljm>

В книге собраны материалы конференции, одобренные Программным комитетом Двадцатой конференции «Свободное программное обеспечение в высшей школе».

Ключевые слова: программное обеспечение, Open Source, свободное программное обеспечение, образование, Linux, цифровизация образования.

УДК 004.91:378
ББК 32.97:74.48

Program Committee:
A. A. Savchenko, — chairman,
A. V. Bondarev, G. V. Kouryachy, A. A. Markina, M. O. Petrova

The Twentieth Conference ‘Free Software in Higher Education’ : conference materials / Pereslavl-Zalessky, February 07–09, 2025. / ex. ed. Cherny V. L. — M.: MAKs Press, 2025. — 144 p.

ISBN 978-5-317-07341-1
DOI: <https://doi.org/10.29003/m4350.978-5-317-07341-1>
EDN: <https://elibrary.ru/njbljm>

The book contains conference proceedings approved by the Program Committee of the Twentieth Conference ‘Free Software in Higher Education’.

Keywords: software, Open Source, free software, education, Linux, digitalization of education.

ISBN 978-5-317-07341-1

© Коллектив авторов, 2025
© Оформление ООО «МАКС Пресс», 2025

Программа конференции

07 февраля, пятница

12.00-15.00 Заселение, обед, регистрация участников

14:30 Автобус от гостиницы «Переславль»

Дневное заседание 15.00–19.00

15.00 Приветственное слово организаторов

15.10–15.40 А. А. Якушин

Современные тенденции использования СПО в Высшей
школе 8

15.40–16.10 С. А. Фомин

«Flip Classroom One More Time» — интерактивность и
асинхронность в эффективных курсах на open-source .. 10

16.10–16.35 В. О. Чайковский

Особенности подготовки свободных учебных
видео-материалов для студентов..... 25

16.35–16.55 Кофе-пауза

16.55–17.25 Г. М. Шефель

О применения СПО в общем ядерном практикуме
Физического факультета МГУ 29

17.25–17.50 В. М. Баканов

Использование метода анализа структуры и
целенаправленных преобразований алгоритмов в
задачах повышения эффективности параллельных
вычислений 32

17.50–18.15	Е. Р. Алексеев, Ю. Р. Чуракова	
	Использование языка программирования Julia на факультете математики и компьютерных наук Кубанского Государственного Университета	36
18.15–18.40	Е. В. Татьянич, П. Б. Жданович	
	Обучение 3D-печати в виртуальной среде	41
19:00	Автобус в гостиницу «Переславль»	

08 февраля, суббота

9:30 Автобус из гостиницы «Переславль»

Утреннее заседание

10.00–13.00

10.00–10.25	Е. Н. Прокофьева	
	Разработка практико-ориентированных заданий в обучении UML-программирования	43
10.25–10.45	Т. О. Зайка, Д. С. Зайка	
	Простота и сложность подготовки рукописи кандидатской диссертации в Libreoffice	49
10.45–11.05	Е. Р. Алексеев, И. А. Горбунова	
	Среда изучения команд терминала Linux	52
11.05–11.30	Д. А. Пилюк	
	Использование СПО в сфере АСУ ТП на примере ОС РВ Embox	55
11.30–11.50	Кофе-пауза	
11.50–12.15	Л. Б. Чашкин	
	Построение моделей цифровых двойников изделий электронной техники	59
12.15–12.40	А. С. Лакиза, В. И. Бондаренко, Д. Д. Лакиза	
	Разработка стенда для обучения студентов системам обнаружения вторжений	65
12.40–13.00	Р. И. Воронин	
	Разработка алгоритмов программного обеспечения для управления отоплением на основе ALT Linux	68

13:00 Автобус в гостиницу «Переславль»

13.00–15.00 Перерыв на обед

14:30 Автобус от гостиницы «Переславль»

08 февраля, суббота

Вечернее заседание

15.00–19.10

15.00–15.30 Л. А. Меркин

Современный C++ для высоконадёжных вычислений в
задачах космической баллистики 72

15.30–16.00 Н. Н. Непейвода

Неполные и неточные задачи в обучении информатике 77

16.00–16.25 А. Г. Михеев, Игумнов О. В

Разработка и использование в Финансовом университете
свободной системы проверки контрольных и
домашних работ на Java 78

16.25–16.50 Н. Е. Шалаев

«Чему учить студентов», или анализ использования СПО в
исследованиях 88

16.50–17.10 Кофе-пауза

17.10–17.35 Д. А. Муканин

Проблемы подготовки прикладных разработчиков в
современной России 91

17.35–18.00 А. С. Речицкий

Как студенту получить первый реальный опыт работы в
крупном проекте на примере ReactOS? 95

18.00–18.25 М. Э. Савин

«Умная игрушка» руками студентов-гуманитариев..... 96

18.25–18.50 Д. М. Султаниязов

Корсаков: человек и российский язык программирования .. 100

19:20 Автобус в гостиницу «Переславль»

09 февраля, воскресенье**Утреннее заседание****10.00–13.30**

09:30 Автобус от гостиницы «Переславль»

10.00–10.25 А. В. Драгунов, А. Т. Нургалиев и др.

Создание облака для сферы образования на базе
свободного программного обеспечения 102

10.25–10.50 П. А. Леляев, К. А. Быховская

Проблемы и решения внедрения СПО в образовательной
организации на примере общеобразовательной школы
г. Москвы 106

10.50–11.15 Д. С. Зайка, Т. О. Зайка

Использование СПО и технологической некромантии в
обеспечении преподавания медико-биологических
дисциплин 111

11.15–11.35 Кофе-пауза

11.35–12.00 И. А. Туманов

Пример настройки АРМ ученика в ОС Альт 113

12.00–12.25 П. А. Бозин

Интеграция работы с памятью переводов в систему
локализации RuBabel 116

12.25–12.50 И. А. Хахаев, П. Г. Петруша

Адаптация пользователей к реальности российского ПО ... 119

13:40 Автобус в Москву

Вне программы

А. В. Филиппов

Использование свободного программного обеспечения для
анализа спектра радиочастот 122

А. В. Живечков

Разработка библиотеки для математической визуализации
из консольных приложений на C++ 126

А. С. Кузьмин

Сложности разработка веб-фреймворка на языке
программирования Python 131

О. М. Семёнова

Обзор инструментов статического анализа уязвимостей в
плейбуках Ansible 134

А. А. Маркина, Д. А. Костюк, А. В. Дубицкий

Практика получения и обработки окулографических
данных при тестировании пользовательских
интерфейсов в GNU/Linux 136

Анатолий ДОС Якушин
Москва

Современные тенденции использования СПО в Высшей школе

Аннотация

Зародившееся в недрах Высшей школы свободное программное обеспечение за время своё существования испытывало как времена бурного развития, так и времена спада к нему интереса в университетской среде. За последнее десятилетие объём и структура СПО в Высшей школе не испытывает серьёзных изменений и практически стоит на месте. Доклад посвящён разбору сложившейся ситуации и обсуждению возможных путей дальнейшего развития.

Ключевые слова: *свободное программное обеспечение, импортозамещении, тенденции развития.*

Свободное и открытое программное обеспечение давно перестало быть уделом небольшой группы маргиналов и превратилось в одну из основных движущих сил развития современной вычислительной индустрии. Исследование *Census III* [1] показывает, что сегодня Open Source-компоненты присутствуют почти во всех современных приложениях, при этом пакеты для облачных вычислений демонстрируют значительный рост такого присутствия, а традиционные модели разработки также быстро эволюционируют в этом направлении. Компоненты с открытым исходным кодом присутствуют в 96% кодовых баз. Подобный вектор развития отрасли требует соответствия от высшей школы в плане предоставления актуальных знаний. Можно наблюдать определённый прогресс в данном направлении, хотя возможно и не столь значительный, как нам бы хотелось.

Исследование, проведённое в прошедшем году АНО «Открытый софт» [2] показывает, что высокая важность обучения студентов культуре работы с открытым и свободным программным обеспечением подтверждается большинством вузов. Можно оспаривать избранную авторами методологию, но в целом результаты соответствуют и другим подобным наблюдениям. Авторы исследования отмечают проблемы и сложности при обучении, разработке, учёте и использовании открытого и свободного ПО, с которыми сталкивается вуз. Чаще всего

это: кадры и компетенции, методическое обеспечение, материальное и техническое обеспечение, реже — требования безопасности и организационные вопросы. Описанные проблемы известны ещё с первых попыток внедрения СПО в вузах, и в принципе за прошедшие десятилетия здесь мало что изменилось.

К сожалению, основная проблема, с которой рано или поздно сталкиваются все, кто пытается внедрить СПО, либо осуществить импортозамещение, состоит в использовании примитивной «механической» методологии внедрения. Суть её состоит в создании неких таблиц соответствия с парами проприетарный-свободный софт и попытками прямо заместить одно другим. В тех случаях, когда пару подобрать не удаётся, делается вывод о невозможности замещения по данному направлению. При этом не учитывается, что свободные операционные системы и их окружение выросли из семейства операционных систем UNIX, где большинство задач решается не использованием громоздкого солидного приложения, а использует совершенно иные подходы.

Сегодняшняя потребность в массовом внедрении СПО позволяет во многом по иному взглянуть на подходы к обучению, провести реверс-инжиниринг учебных и организационных процессов, привести их к соответствию unix-way методологии.

Литература

- [1] Frank Nagle, Kate Powell, Richie Zitomer, David A. Wheeler, *Census III of Free and Open Source Software. Application Libraries.*, December 2024
- [2] АНО «Открытый код» *Отчёт по исследованию применения в вузах открытого и свободного программного обеспечения и обучению культуре работы с открытым кодом*, 2024 URL: <https://russiaaos.ru/news/174/>

Стас Фомин
Москва, ИСП РАН, МФТИ

«Flip Classroom One More Time» — интерактивность и асинхронность в эффективных курсах на open-source

Аннотация

Обучение в ВУЗах до сих пор в целом средневековое, несмотря на конкуренцию с глобальным интернетом. Но «Восстание ИИ» добивает стандартные задачи и экзамены!

С прошлого тысячелетия автор непрерывно экспериментировал в преподавании нетривиальных курсов и пришёл к развитию принципов *flipped classroom*:

- Интерактивность материалов.
- Совместная работа в браузере.
- Асинхронное взаимодействие вместо лекций-семинаров.
- Одноразово-индивидуальные практические задания, превращающиеся после решения в обучающие материалы.

Оказалось, это просто реализуемо с свободным софтом — достаточно запустить `code-server` на персоналке. Получатся «интерактивные образовательные квесты», причем курсы смогут расти и сами, с помощью студентов.

Ключевые слова: *дистанционное образование, интерактивные курсы, геймификация образования.*

Введение

Несмотря на технический прогресс, который мог бы давно улучшить образование в ВУЗах, там до сих пор всё сковано средневековыми подходами — «лекции в аудитории с доской» и «выматывающие учителя и студента семинары по расписанию», «книги в библиотеке» и «набор древних задач со давно слитыми решениями», не говоря уж про «театр потоковых экзаменов», где разыгрывается унылая пьеса про «справедливую оценку знаний», приводящая с нервному срыву как студентов, так и экзаменаторов. Да, лекционный схоластический формат родился в глубоком средневековье, когда профессор был тот кто «написал свою книгу». Студенты — те, кто слушали, как он её

читал, и конспектировали. Набор конспектов и был их «свидетельством образования». Тогда это как-то работало, но сейчас уже нет — материалы доступны, студенты не видят смысла в записи-конспектировании-переписывании, хотя основной смысл был именно в процессе осознания, а не в результате-копиях. В результате «знания» сводятся к талмудическому зазубриванию фактов-шаблонов и элементарным символическим действиям «ручкой на бумаге», не давая практических навыков применимых в современной индустрии. А изменить процесс часто не дают бюрократы, сковывающие учителя нормативами лекционных часов и семинарских занятий.

Конкуренция с онлайн-курсами и интернет-ресурсами запустила изменения, но недостаточные, чтобы сломать эти «славные традиции». Хороший пинок дала ковидная истерия: появились массово видеозаписи «лекций у доски» и попытки созвонов вместо семинаров. А в этом году, взлёт больших языковых моделей, известных в народе как «нейросети»TM, привёл к плачу учителей по всему миру «что делать?» — «студенты не слушают меня, а смотрят англоязычное в нейропереводе», «ИИ решил все мои задачи». Задумаемся о другом — «что не делать?» — чтобы было время и силы на что-то более полезное, чтобы у студентов были практические навыки и уверенность в понимании.

У меня ежегодный опыт «с прошлого тысячелетия» курсов по алгоритмам-криптографии и подобному Computer Science в МФТИ и ИСПРАН. Прошёл эволюцию от «доски с мелом» и «дуэлей на устных экзаменах», через книги и лекции с слайдами, к прогрессивным форматам, удобным и эффективным студентам и преподавателям. Всегда использовал только свободный софт, а то, что предлагаю тут, легко развернёт любой линуксоид даже на персоналке.

Flipped Classroom и интерактивная коллаборация в браузере

Десятилетия я пробовал и продвигал две ключевых идеи:

- *Flipped Classroom*, когда всё лекционное готовится заранее, книгами-слайдами, а главное — компактными видеороликами, спасающими от требования, что «курс должен быть прочитан». Благо свободный OBS даёт всем «видеостудию» не вставая с любимого стула [5], а копеечный планшет-стилус [1] позволит не

тосковать о «доске с мелом» и других доисторических артефактах.

- Работы с студентами в какой-то общей, дистанционно доступной среде, не требующей от студента ничего, кроме браузера — начиная от MediaWiki (см. [2], [3]), эволюционируя до сервисов коллаборации, таких как Cocalc, JupyterLab, CodeServer [6] — чтобы можно было работать не только на очных семинарах, но и дистанционно, без «пера и бумаги».

В результате, я пришёл к следующим выводам:

- Самая удобная среда для совместной работы на ближайшее десятилетие — **code-server**, т.е. «VSCode в браузере» — самый стандартный интерфейс, который или уже знаком студентам, или скоро им понадобится в любой IT-профессии.
 - Развернуть его в пару команд можно любому айтишнику даже на домашней персонлке, пробросив **ssh**-туннель до любого облачного сервера.
 - Его возможностей хватит не только на «юпитер-ноутбуки с питоном», но и для полноценного программирования с отладкой практически во всех стеках, документирования и ведения обучающих материалов — есть тексты-формулы, рисование схем, всё необходимое.
- Не нужно думать о «публикации обучающих материалов», мучаясь с мудреными системами сборки PDF-книг и слайдов по LaTeX коду, или там системы публикации книг-сайтов типа Quattro или Sphinx/RST. Это конечно тоже возможно! Но не нужно!
- Пусть студент и преподаватель наравне работают — изучают, правят, экспериментируют в одной и той же среде, и обычно достаточно того, что может показать сам **Code-Server**
 - jupyter-ноутбуки, markdown-документы, drawio-схемы, код;
 - там могут быть интерактивные и анимированные штуки;
 - картинки, видео и работающие системы симуляции;
 - даже простые Markdown-документы с мгновенным просмотром включают и формулы, и Mermaid-диаграммы, есть множество расширений, позволяющих создавать конспекты, наброски статей, задачи — без стандартных мучений с TeX и текстовыми процессорами.

Лично я экспериментирую с изобретенным форматом «бесслайдовых» конспектов-майндмапов, которых можно создавать «со скоростью мысли», на порядки легче, чем верстка в LaTeX/Beamer, не говоря уже о всяких поверпоинтах. Ведь достаточно осознать, что «слайдовая разбивка» нужна только несчастному докладчику с примитивной листалкой. Если вы что-то рассказываете-показываете не отходя от клавиатуры — у вас на порядок больше возможностей навигации, и важно сконцентрироваться на выделении и структуризации ключевых идей (чтобы остальное проговорить голосом) — а это можно хитро оформить CSS-стилями для просмотра Markdown-документов.

Для демонстрации «а что, так можно было?», я создал в этой среде не только разные курсы по вычислительной математике, алгоритмам и теории сложности [14], [15], [13], [17], но также спецкурс по физике/аналитической механике [18], а чтобы показать, что Jupyter-ноутбуки не только «про питон» — интерактивный курс по изучению Haskell [16].

Flipped Asynchronous Seminars

С этим арсеналом я попробовал атаковать следующее после лекций наследие темных веков — семинары. Падение интереса к семинарам в «глобальную информационную эпоху» отмечается многими [8], но у автора был личный вызов — «курс по выбору для шестикуров МФТИ» на полсотни студентов, где всё было особенно сложно — эти студенты уже давно плотно работают, в массе не посещают аудиторных занятий, их оказалось нереально собирать даже на дистанционные семинары-созвоны. Причем многие «втягиваются» только спустя пару-тройку месяцев, а то и вовсе после официальной даты экзамена. К сожалению, тупо послать немотивированных нельзя, нужно принимать переэкзаменовки, чем эти студенты и пользуются («ну давай *уд*, а то всё равно замучаю»). Да, это был не совсем курс по выбору — выбор был, но альтернативой были совсем странные, малопрактичные и непопулярные курсы, от которых был поток немотивированных «беглецов». Это конечно проблема процесса, но в этом был и некий вызов — как «интегрировать в работу» и таких «беженцев», что было нереально, если придерживаться фиксированных планов семинаров.

И решение пришло: давайте *flip*-нем и общепринятые **гендерные** учебные роли! Вместо стандартного процесса «избитый набор задач, часть разбирается на семинаре, остальное студенты списывают друг

у друга», поднимем студента до преподавателя! Пусть студент тоже творит обучающие материалы, решая индивидуальные задачи! Использует те же технологии, что и преподаватель, создавая код-текст-юпитер-ноутбуки и объясняющий видеоролик!

Это разом убивает кучу зайцев:

- *Нет проблем со списыванием* — подсматривайте решения друг друга, помогайте, среда и процесс за это!
- *Мотивация* — это не одноразовая работа на выброс, сдал-выкинул-забыл, почти нетленка, обучающий материал, хотя и некристичный, если будут ошибки или косяки. Конечно, делаются пара итераций, пре подаватель правит ошибки студентов, но и студенты «отлаживают» материалы учителя!
- Курс органично «растет вширь» без даже преподавателя!
- Студенты тренируют навыки публичных выступлений — гораздо эффективней, чем просто давать выступить на семинаре. Это полезно как для подготовки защиты диплома, так и профессиональной или научной работы. Перевернув «конспектирование лекций» мы снова добиваемся чтобы «понимание» бегало между аналитическими и разговорными центрами.
- Практические навыки «быстрых видеоотчетов» востребованы в современном IT, ну а техническое видеоблогерство — полезная рыночная ценность.
- И всё это асинхронно — можно начать работу в любое время, хотя конечно, надо поощрять первопроходцев.

Задания стоит делать мотивирующими:

- «вы математик-аналитик, решающий бизнес-задачу до прототипа приложения»;
- «вы преподаватель или видеоблоггер, разбирающий алгоритмическую задачу для собеседования в IT-компанию»;
- «вы исследователь, рецензирующий статью, или готовящий семинар по ней»;
- ну или подобным образом, чтобы студент чувствовал «получение полезных навыков и уверенности».

«Навстречу» студентам идут разборы решений от преподавателя. Преподаватель, если хороший профессионал, сильно загружен настоящей работой в науке или индустрии, но имея общую среду, может заняться разбором решений без всяких «резервирований времени, подготовок к семинарам, путешествий к аудиториям» (страшно вспом-

нить свои поездки в долгопу, по два часа в каждую сторону, возню с свертыванием-развертыванием проектора, и всё ради часового занятия), в любое выдавшееся свободное время при минимально вменяемом ментальном состоянии.

Технически, «подключение» занимает минимальное время:

- Постоянно открытая вкладка в браузере с проектом (или несколькими вкладками) — браузеры научились помнить контекст, и у вас всегда проект под рукой, преподаватель всегда может посмотреть — что там происходит в курсе?
- Запуск OBS, просмотр решения студента и видеозапись экрана в режиме Pause/Упраuse (чтобы скрыть паузы на обдумывание). Исправление ошибок-косяков, показ лучших практик, и параллельно быстрое составление конспекта ключевых проблем и советов.
- Для конспектов я также применял метод «бесслайдовых mark-down-майндмапов» приправленных емодзи для вдохновения — это когда-нибудь будет темой отдельного доклада.
- Ура, готов миниблок-разбор-минисеминар!

Кстати, подобная ситуация и у студентов, и им тоже легко включаться в работу «по возможности».

Этот взаимный поток решений и разборов можно:

- просто записывать в каком-нибудь файле **feedback.md**;
- вести чаты-группы-топики в ТГ-каналах
 - туда можно забрасывать и конспекты разбора и ссылки на видео, или просто двухстрочные замечания «так хорошо / так не надо»;
 - можно дублировать это и студентам в личку;
- видео можно выкладывать на видеосервисы или в любое файлохранилище (стандартное облачное, хранилище университета, nextcloudы, FTP, сайты), сейчас любой браузер проиграет обычный MP4-файл, где был он ни был, без нужды в каком-нибудь хитром ю/ру-тубе;
- это и будет «асинхронный непрерывный семинар», который сохранит свою актуальность для следующих семестров — его смогут смотреть новые поколения студентов!
- формат максимально компактный и эффективный — только решения студентов и разборы ошибок, «проиндексированный конспектами» — несравнимо по эффективности по сравнению

с ужасно унылыми записями многочасовых «мастерклассов в аудитории», где большую часть времени происходит непонятная возня.

Некоторое количество созвонов сохранилось, но это было в основном общение 1:1:

- Для решения какой-то конкретной проблемы-блокера минут за пять — легко договорится «не понимаю почему не работает — ладно, покажите, го в созвон?»
- Либо долгие созвоны при разборе какой-то исследовательской задачи (разбора статьи, глубокого оптимизационного исследования и т.п.) — для тех, кто занимался «индивидуальным глубоким погружением» — и именно в режиме ненапряжного «парного программирования», в отличие от групповых, созвоны максимально эффективны.

Кстати, парные созвоны оказалось удобно делать даже в ситуациях, когда студенты были на работе в организациях, где забанены WebRTC-протоколы, на которых работает весь софт групповых созвонов (а такое всегда можно ждать от РКН) — тогда можно созваниваться через телефон (телеграмом или просто), и совместно что-то делать в общей среде.

Надеюсь кто-то из этих студентов станет преподавать по-новому, ведь смена научных и технических парадигм окончательно происходит только *со сменой поколений носителей*.

Восстание Машин

Отдельно стоит отметить, что подобный подход устойчив к «Проблеме Года»™ — роста разума больших языковых моделей, рожающих гладкие разумные тексты и решающих среднесложные задачи. Многие знакомые преподаватели жаловались, что «хакнуты» их экзамены с классическими наборами задач, не говоря уж о тех, кто в качестве работ принимал тексты. Ценность «гладко написанного» текста на любом языке упала до нуля, дистиллированные математические и алгоритмические задачки щелкаются *чатгопотой* на ура. Из личного общения со студентами, узнал, что практически у всех, несмотря на казалось бы сложность зарубежного платного доступа, есть доступ к самым ведущим моделям, на конец 2024 — `chatgpt4o`, хотя и условно-бесплатный `deepseek` тоже демонстрируют

неплохие результаты. Лично наблюдал, как нейросетями были «перебиты» мои старые наборы теоретических и алгоритмических задач. Да, мой старый подход — давайте для разминки порешаем задач с «leetcode/spojcode/codechef» — уже не работает.

Хорошо это или плохо? Плохо, когда разум студента совсем в этом не участвует, когда телефон фоткает задачу и копируется ответ. Но выкидывать совсем? Это ведь ценный современный инструмент, и нужно учиться им пользоваться — от навыков *prompt engineering* до декомпозиции задач!

Скорее стоит подумать «учить ли логарифмическим линейкам в компьютерном мире?» © Полезно ли зубрить доказательства теорем, если люди не в состоянии применить это к реальности? В индустрии автор часто встречал выпускников ведущих ВУЗов, прослушавших курсы оптимизации с десятками теорем про сходимос-ть-оптимальность-базисы-краевые условия... и при этом не способных сформулировать и даже понять постановку задачи оптимизации для их проблемы! «Поисковая эрудиция» уровня «синтаксиса языков» и «API библиотек», «зубрежка буквоедских определений» и «вывод простых формализмов» — это уже не показатель крутизны специалиста, это всё заберет «ИИ», а вот «сопряжение формализмов с реальностью» — становится всё более и более важным.

Удивительно, что именно Болливуд выпустил «Три идиота» — один из лучших фильмов про проблемы современного образования, стоит посмотреть там хотя бы эту сцену про бессмысленность зубрежки формальных определений [11].

Надо ставить такие задачи, чтобы они сразу не решались *чатгоп-той*, но совершенно нормально, если ИИ будет что-то подсказывать студенту, пусть он занимается с ним маевтикой без преподавателя!

Например, простые алгоритмические задачки — да, ИИ может решить.

- но надо сделать не просто решение, с русификацией и максимальной читаемостью, а визуализацию работы алгоритма (см. [7]):
 - с использованием редкого фреймворка, в котором нужно еще разобраться;
 - с хорошими визуальными метафорами;
 - и о котором нейросети ничего не знают, и вряд ли будут.
- и еще записать видео с разъяснениями «как оно работает»...

Или надо поставить формулировку матпрограммирования по бизнес-задаче...

- но с использованием специальных редких модулей-хелперов для моделирования данных;
- с специальными требованиями по формулированию модели;
- с получением бизнес-приложения в одной ячейке;
- и опять с видеоразбором.

Опробовано еще куча идей — «вот простые задачи по CS/математике, но решать их надо через *sympy*», нет места перечислять их все, но я думаю, смысл ясен.

Возможно когда-то ИИ сможет и это всё, но сложность и количество промтов к нему будут сильно больше, чем просто разобраться в задаче и сделать всё самостоятельно.

Отдельный открытый вопрос про быстрые тесты с простыми вопросами [2], польза от которых тоже бывает, и как их сделать малоуязвимыми к быстрой атаке без промтов («телефон с фото») — это вопрос пока открытый, но интересный, есть идеи, возможно им стоит заняться.

И конечно надо прекратить в ВУЗах заниматься предметами, обучение которым можно тупо автоматизировать еще на уровне школьников даже без ИИ — всякие «основы питона» и прочая элементарщина, не говоря уже про распространенных в РФ вредительских курсах по изучению «самодельных языков программирования». Тратить ресурсы на это просто недопустимо [4].

Что нужно и что не нужно готовить для интерактивного современного курса

С таким подходом не нужно писать книги, особенно большие и толстые. Во-первых, они скорее всего уже написаны, и лучше, чем вы сможете. Возможно на другом языке, но теперь это неважно, почти все книги доступны (спасибо ZLibrary), а прогрессирующие технологии автоперевода (спасибо ИИ) снимут ценность «ну зато же на русском-татарском-армянском...». И да, «публикация на бумаге» — давно абсолютная бессмыслица.

Во-вторых, книги своей толщиной защищают себя от прочтения, герметичная внутренняя структура книг с собственным набором определений и внутренних ссылок делает эти монолиты негибкими.

Надо делать материалы максимально нарезанные на атомы (конспекты, юпитер-ноутбуки, код, всё разложено по папкам, связано ссылками), каждый из которых можно изучать самостоятельно и в рамках разных курсов. Придерживайтесь правил «независимости и простоты» это будет удобно читающему:

- Все определения — прямые ссылки на википедию или другие авторитетные сайты по теме.
- Все ссылки именно напрямую, гиперссылками, без дурацких списков литературы и библиографий — если есть нужная книга-статья — должна открываться по клику, или «ненужно».
- Все иллюстрации — не «ищи где-то Рис. 1488», а максимально по месту, ведь если у вас нет страничной верстки, не нужно экономить бумагу. Можно повторить картинку сколько угодно раз в разных контекстах, если необходимо.
- Аналогично про формулы-уравнения-таблицы, и т.п. — просто повторите, не заставляйте куда-то лезть, всё важное для каждой мысли пусть будет на одном экране.
- Если возможно, спрятать унылые выводы с переписыванием и упрощением формул в системы компьютерной алгебры, типа *sympy*. Зато основные идеи и каркасы доказательств хорошо бы визуализировать схемами.
- Лишний текст стоит убирать, если он не помогает — он только отнимает место и «очки внимания» у важного.
- Использовать возможности форматирования, цвета, анимации, картинок, и даже эмодзи, для максимальной активации восприятия (не будем тут начинать про «правое-левое полушарие», но это работает!).

Не нужно записывать многочасовых видеолекций! Вот пугающий пример курса из 100 (ста!) лекций по программированию на Haskell [10], не говоря уже о ужасном видеокачестве с мерцающим проектором, где только на шестой лекции начался разговор собственно о языке, а не о таких интересных вопросах «какие крутые в СССР были академики», «могут ли девушки учиться» — не надо так. Если и делать — коротко, максимально круто и визуально, равняясь на видеоканалы уровня «3blue1brown» [9]. А если не получается так — просто дайте студентам ссылку на лучшее, пусть студенты смотрят с автопереводом, не надо «импортозамещение локализацией».

Но нужно делать то, с чем и будут «сражаться» студенты — интерактивные материалы, тесты, задачи, симуляции. И вот ради поиска таких задач нужно продолжать рыть все возможные книги по вашей теме, вытаскивать интересные примеры, и оформлять их задачами — и знать, что рано или поздно они станут «разобранными примерами». Это будет поддерживать в вас тонус, и наводить на интересные мысли — как сделать лучше. Например, автор статьи при построении курсов прошерстил и прорешал задачи десятков книг (и нашел множество ошибок).

Иногда задачи можно искать на открытых сайтах (алгоритмические и математические задачи), но тут надо так их подать и переформулировать (см. выше), чтобы не срабатывал тупой поиск решения, ведь решения скорее всего всем известны.

Ну и всегда, для продвинутых студентов можно предложить квест по анализу свежих статей в их области на arxiv.org, с конспектированием и переработкой, с реализацией и проверкой алгоритмов или результатов — это и отличный вызов, и важная миссия верификации, и хороший шанс написать статью с опровержением или каким-то улучшением — отличный путь в науку или на передовой край технологий.

Конечно, всё это подразумевает минимальную вменяемость студентов, возможно малоприменимо условно на «первом курсе» стандартного ВУЗа, где всё еще ликвидируют базовую неграмотность студентов в элементарных школьных предметах, читают сверхклассические курсы типа «матан 101», и по сути, пока идет «курс молодого бойца» и послеекзаменационный доотбор.

Долой экзамены

Важный результат такого «асинхронно-практического» подхода — ненужность классических потоковых экзаменов, которые и так бессмысленны для нетривиальных предметов.

Фантастична сама идея, что знания современных технологий достоверно оцениваются за час, да еще и на потоке. Тогда бы закрылась HR-индустрия с кучей собеседований, тестовыми заданиями и испытательными сроками. Формат экзамена диктует «искать где светло, а не где потеряли» — спрашивать то, что можно легко и быстро проверить: какие-то определения, простые факты и элементарные задачи.

Но это всё — рутина и зубрежка, то, что скоро будет автоматизировано.

При этом, «синхронный экзамен», особенно устный — дичайший стресс как для экзаменатора, так и студента, приводит к конфликтам уровня «студент послал преподавателя», «не повезло, преп был уставший, не стал слушать, не повезло». А ведь за пределами базовых курсов, где специально обученные дрессировщики крутят годами свою шарманку, обычно и студент и преподаватель — специалисты, которым потом с большой вероятностью придется вместе работать, и зачастую, тоже в *Flipped Position* (да, я не однократно работал с/под своими студентами).

Описанное практически-ориентированное асинхронное обучение позволяет студенту всегда видеть «набранные баллы» и регулировать вклад, в соответствии с ожиданиями, интересом к курсу, и имеющимися ресурсами. Делать выбор — семья-работа-бизнес-учеба — сколько угодно, только это уже проблема студента, его рационального выбора без всяких лотерей! Да и можно сказать, что устный экзамен тоже состоялся — преподаватель и студент немало видели и слышали друг друга, хоть не встречались лично и возможно живут в разных часовых поясах или циркадных ритмах.

Преподаватель только задает «правила игры», помогает и следит за «игровым балансом» — чтобы было не слишком просто или сложно, чтобы не было откровенного «читинга», и главное — чтобы все занимались обоюдополезным, и «мир становился лучше от каждого вклада студента или преподавателя».

Homeless Serverless стартап

Часто возникает вопрос — ну я как бы преподаватель, но хочу попробовать без возни с виртуалками и вообще превращения в *красноглазого* или *девопса*, а тут вроде обещано, что в пару строчек всё можно запустить, и хотим попробовать вот небольшой группой. . .

Okay, раньше действительно я развертывал сложные коллаборашки типа Cocalc в облаках, с использованием бестипрактик DevOps (Terraform/Ansible/Pyinfra...) но с активным использованием `code-server` тоже перешел к «бомжшаредхостингу» — по сути, несколько десятков образовательно-исследовательских проектов живут в миниатюрном Mini-ITX десктопчике, который стоит у меня на столе в институте, еще несколько — на древних домашних компах, т.к.

в Mini-ITX не влезали графические карты, нужные для машинлернинга и нейросетей, купил их б.у... Как ни странно, это оказалось более эффективно (память-ядра-диски-аптайм-айпиадреса...), чем через доступные мне в институтском облаке виртуалки.

Хорошо, будем считать, что хоть линукс на вашем компе-ноутбуке есть.

- Поставьте **code-server** — он может быть опакечен в вашем дистрибу, ну или зайдите на <https://github.com/coder/code-server/releases>
- Заведите отдельного пользователя под проект, где вы будете работать группой, запустите сервис под ним, и профорвардите порт, ну вот те самые три строчки для этого:

```
sudo useradd oseducnf
sudo systemctl enable code-server@oseducnf --now
sudo -u oseducnf ssh -R 80:127.0.0.1:8080 nokey@localhost.run
```

В консоли увидите адрес типа `1a0a0d305040dc.lhr.life`, заходите на него по **http**, пароль найдете в `/home/oseducnf/.config/code-server/config.yaml`. Делитесь этим секретным адресом и паролем в вашей секретной ТГ группе и приступайте к совместным проектам! Зайдите там в плагины, наставьте всякого полезного, начинайте работать. Получили «советский сервер», включенный только в рабочее время на вашей персоналке!

Это конечно не очень секьюрно, но про всё остальное (TLS-терминацию, покупку доменов, виртуалки о облаке для белого айпи, **certbot**, SSH-форвардинг...) — спросите ИИ, начните с промпта «code-server config.yaml how to https» у хотя бы chat.deepseek.com, придерживаясь изложенных принципов не будем тратить бумагу на тривиальное, это как раз работа для ИИ.

Литература

- [1] Фомин, С. А. *Магия пера или эффективная свобода преподавания со стилусом*, // OSEDUCONF-2014 // Девятая конференция «Свободное программное обеспечение в высшей школе» : Тезисы докладов, Переславль, 25–26 января 2014 года. — Переславль: Альт Линукс, 2014. <https://0x1.tv/20140126-4>
- [2] Фомин, С. А. *MediaWikiQuizzer или ВикиЭкзамены — тесты, удобные и для преподавателя и для студента* // OSEDUCONF-2015 // Десятая

- конференция «Свободное программное обеспечение в высшей школе»: Тезисы докладов / Переславль, 24–25 января 2015 года. — М.: Альт Линукс, 2015. — 100 с. : ил. <https://0x1.tv/20150125K>
- [3] Фомин, С. А. *Эффективная «домашка» — задачи студентам на MediaWiki*, // Свободное программное обеспечение в высшей школе : тезисы докладов, Переславль-Залесский, 2017 года / НОУ «ИПС-Университет г. Переславля им. А. К. Айламазяна», Институт Программных Систем РАН, Институт Логики, Базальт СПО. — Переславль-Залесский: Basealt, 2017, <https://0x1.tv/20170129D>
- [4] Фомин, С. А. *Udaff — русский пиктографический Python. От элементарных алгоритмов до гомоморфного шифрования* // Свободное программное обеспечение в высшей школе : Сборник тезисов XV конференции, Переславль, 07–09 февраля 2020 года / Отв. редактор В.Л. Чёрный. — Переславль: ООО «МАКС Пресс», 2020. — С. 121–127. — EDN JZBFRI. <https://0x1.tv/20200208Q>
- [5] Фомин, С. А. *OBS — швейцарский нож передачи знаний. Боевые приёмы Open Broadcaster Software* // Свободное программное обеспечение в высшей школе : Сборник тезисов Четырнадцатой конференции, Переславль, 25–27 января 2019 года / Ответственный редактор В.Л. Чёрный. — Переславль: ООО «МАКС Пресс», 2019. — С. 82–92., <https://0x1.tv/20190126Q>
- [6] Фомин, С. А. *Современные «интерактивные среды» и «живые лаборатории» — эффективное дистанционное образование по алгоритмам и математическим дисциплинам* / С. А. Фомин // Восемнадцатая конференция. Свободное программное обеспечение в высшей школе : Тезисы докладов материалов конференции, Переславль-Залесский, 27–29 января 2023 года / Отв. редактор В.Л. Чёрный. — Москва: ООО «МАКС Пресс», 2023. — С. 63–64. — EDN GIZTTL., <https://0x1.tv/20230128F>
- [7] Фомин, С. А. *PyAlgovizualizer — эффективное преподавание алгоритмов* / С. А. Фомин // Девятнадцатая конференция «Свободное программное обеспечение в высшей школе» : Материалы конференции, Переславль-Залесский, 28–30 июня 2024 года. — Москва: ООО «МАКС Пресс», 2024. — С. 69–75., <https://0x1.tv/20240629H>, <https://gitlab.ispras.ru/discopal/pyalgovizualizer>
- [8] Курячий, Г. В. *Кризис «средневековой» модели образовательной площадки в условиях информационной связности* / Г. В. Курячий // Девятнадцатая конференция «Свободное программное обеспечение в высшей школе» : Материалы конференции, Переславль-Залесский, 28–30

июня 2024 года. — Москва: ООО «МАКС Пресс», 2024. — С. 11–14. — EDN MVJFTI. <https://0x1.tv/20240628B>

- [9] *Прекрасные примеры визуализации при разборе математических востроров и IT технологий*, <https://www.3blue1brown.com>
- [10] «Курс: Программирование на Haskell», <https://altube.ru/channel/intuit/playlists/programmirovanie-na-haskell>
- [11] *Одна из ключевых сцен в фильме «Три идиота» о проблемах традиционного образования, о бессмысленности зубрежки формальных определений*, <https://www.youtube.com/watch?v=-MlkASchodc>
- [12] Проект MediaWiki4IntraNet, в котором есть множество расширений, полезных для использования MediaWiki в качестве LMS, и используемой автором доклада, <https://wiki.4intra.net/>, <https://github.com/mediawiki4intranet>
- [13] Курс «Визуализация алгоритмов», <https://discopal.ispras.ru/Algo-vusial>, <https://gitlab.ispras.ru/discopal/algo-visual>
- [14] Курс «Моделирование бизнес-задач», <https://discopal.ispras.ru/Business-modeling-pyomo>, <https://gitlab.ispras.ru/discopal/adv2022-course-pyomo-business-optimization>
- [15] Курс «Моделирование труднорешаемых задач», <https://discopal.ispras.ru/Hard-problem-modeling>, <https://gitlab.ispras.ru/discopal/hard-problems-formulations>
- [16] Курс «Интерактивный Haskell», <https://discopal.ispras.ru/Haskell-exercism>, <https://gitlab.ispras.ru/discopal/haskell-exercism>
- [17] Курс «SymPy для алгоритмов», <https://discopal.ispras.ru/Sympy4algorithms>, <https://gitlab.ispras.ru/discopal/sympy4algorithms>
- [18] Спецкурс по аналитической механике с SymPy, <https://discopal.ispras.ru/Mechanics-sympy-intro>, <https://gitlab.ispras.ru/discopal/mechanics-sympy-intro>
- [19] <https://gitlab.ispras.ru/discopal/matplotlib-intro>
- [20] <https://gitlab.ispras.ru/discopal/css-md-vscode4show>
- [21] https://gitlab.ispras.ru/discopal/pyomo_helpers
- [22] https://gitlab.ispras.ru/discopal/py_svg_combinatorics

Виталий Чайковский
Мытищи, МГУ им. М. В. Ломоносова

Особенности подготовки свободных учебных видео-материалов для студентов

Аннотация

Мы все живём во время развития и потребления видео-контента. Образовательного материала на разных площадках много. Кроме того, зрители в своей массе привередливые, увлечь их сложно. Сделать что-то уникальное в области образовательных роликов ещё сложнее.

Но, нам, с нуля, удалось создать канал, на котором, Г. В. Курячий вот уже много лет успешно читает лекции из МГУ. Канал развивается, регулярно привлекает новых подписчиков *без финансовых вложений и рекламы*.

Как так получается, я постараюсь рассказать в своём докладе.

Ключевые слова: *видеомонтаж, композитинг, звукозапись, съёмка*.

Оборудование для видеозахвата и звукозаписи проводимых лекций

Короткая секция. В ней будет представлена информация обо всех устройствах, используемых непосредственно при записи и проведении лекций.

Особенности захвата. Предупреждение возможных проблем. Как можно улучшить в будущем. Техническая часть

В чём идея? Чем обусловлен выбор оборудования. Настройка, оптимальные параметры. Проблемы, возникающие во время записи и трансляции и их возможные решения. Вся предварительная подготовка перед монтажом будет описана здесь.

Видеомонтаж, композитинг. Творческая часть. Воспроизводимость

Видеомонтаж лекций задача времязатратная, но, кажется, довольно не сложная и тривиальная. Однако, я поставил перед собой цель

смонтировать лекцию так, чтобы впоследствии, она выглядела как единое целое, без явных склеек и переходов. Это уже не просто. Нужно применить не только знания видеомонтажа, но и некую долю изобретательности.

Нужно сделать так, чтобы зритель словно бы сам присутствует на лекции, и у него не возникнет ощущения, что какая-то информация удалена после монтажа. При этом из лекции, на самом деле, выкидывается очень много мусора. Начиная от косяков самого лектора и заканчивая, изредка случающимися, происшествиями во время записи. В монтаже от них не остаётся и следа — так, словно ничего этого и не было. Лишь присутствовавшие очно студенты, возможно, сохраняют эти события в памяти. А вот зритель о них, мало того, что никогда не узнает, так ещё и не подумает об их возможном существовании.

Такой дотошный подход, конечно, отнимает намного больше времени, чем обычный «ленивый» монтаж. Это похоже на настоящий, сложный пост-продакшен. Иногда, приходилось править очень серьёзные косяки. Например, задействовать вместо лектора «живой» 3d аватар. Приходилось не раз устранять наличие троллей в чате трансляции. Всё это никогда не входило в мои обязанности, но как можно было оставить лекцию без говорящей головы или мусором в чате? Поэтому я считаю, что всё сделанное в рамках, и вне рамок моей работы того стоит.

В моей работе, в основном, используются простые, но эффективные приёмы. Их будет не сложно исполнить в любом редакторе нелинейного монтажа, не привязываясь к конкретному ПО. Это значит, что всё то, что я описываю выше, воспроизводимо кем угодно. Всё, что нужно — владеть любым удобным редактором нелинейного монтажа на базовом уровне, знать некоторые азы операторской работы, иметь небольшой творческий потенциал и понимание того, что должно получиться в итоге.

Я, конечно же, не обладаю знаниями возможностей всех подобных ПО, но о том, какой функционал должен быть в нём заложен, чтобы воспользоваться освещённой мной информацией — отмечу.

Подача лекций. Немного практической психологии

Не секрет, что сухое изложение материала многим воспринимать сложно. А человек учится постигать многое, играя. И я сейчас вовсе не про детские или компьютерные игры. Речь идёт о подходе в по-

даче материала. Здесь, конечно, не малую роль играет как харизма лектора, так и специфические вещи, включённые нами в лекции.

Например, уникальные заставки каждый новый сезон, придают роликам немного привлекательности. Бывают отдельные специальные заставки на некоторые популярные в народе даты. Изредка, в изложении проскакивают мемы, шутки, какие-то мимолётные приколы. Их не очень много. Они, скорее, как «пасхалки» к лекциям, чем регулярный юмор. В конце-концов, у нас учебный курс, а не юмористические ролики. Но их наличие всё же играет роль, причём не одну. Одна из таких — расположение людей, настройка их на нужный лад. Это объединяет нас со зрителями, сообщая им о том, что авторы канала не оторваны от народа. Что мы на одной волне. Это полезно как для восприятия материала, так и для привлечения новых подписчиков.

Для улучшения качества лекций так же не обойтись без банальной цветокоррекции, шумодава и т.п., но это больше к технической части относится, чем к творческой.

Замечу, что канал за время своего существования набрал свыше десяти тысяч подписчиков, не покупая рекламу и не рекламируя себя. Это вполне неплохой результат для канала, цель которого доносить знания, а не увеличивать охваты и привлекать народ.

Доля студентов, приходящих на лекции очно, явно составляет небольшой процент от общего числа подписчиков. Накруткой и ботами мы не балуемся. Получается, что к нам пришло как минимум девять тысяч человек, благодаря работе т. н. «сарафанного радио». Люди часто хвалили в комментариях и чате харизматичного лектора, интро, монтаж, редкие мемы.

Благодаря этим отзывам, можно сделать вывод, что мы двигаемся в правильном направлении.

Здесь же можно ответить на вопрос — чем наш лектор лучше других, если подобных курсов в интернете масса.

Скажу честно — платят мне ровно за монтаж и съёмки. Заставки и прочие технические изыски — лишь моё личное желание сделать всё ещё лучше и подкачать собственный опыт. Например, для замены лица Георгия на одной из лекций мне пришлось провести самостоятельное, довольно длительное изучение темы захвата мимики лица и переноса его на виртуальный аватар. Такие, вещи, конечно, не обязательны, но просто интересны и могут вдохнуть больше жизни в наш

проект в будущем. К слову, такое тоже воспроизводимо, но требует предварительного изучения дополнительных инструментов.

Популяризация СПО

Не могу не отметить то, что на самих лекциях часто идёт речь о свободных программных продуктах. На подобных инструментах делается акцент, уделяется внимание разработке с их использованием, свободным операционным системам и тому подобным вещам. Лектор сам использует свободное ПО в эфире и не раз об этом говорит.

Мы популяризируем СПО в широких кругах, среди всех наших зрителей и слушателей. К слову, порой, в онлайн на лекции к Георгию заходят не только поучиться, но и просто послушать его фоном. Странное, казалось бы, дело. Но, фактически, даже такие зрители есть. Остаётся надеяться, что любители такого странного «asmr» хотя бы что-то воспринимают.

При помощи монтажа, кстати, так же можно сфокусировать внимание на чём-то конкретном. Верно, мы можем использовать свои навыки не только для удаления информации, но и для её подчёркивания.

Отдельно отмечу тот факт, что последние годы, мою работу спонсирует компания Базальт СПО. Потому на роликах можно наблюдать их логотип, их интро в начале и контактные данные в конце каждого ролика. Возможно, это даже смотивирует некоторых будущих выпускников МГУ и других людей обратить внимание на компанию, чтобы сотрудничать с ней в дальнейшем. Кстати, один из последних курсов Георгия, параллельно Ютубу, заливался на портал с курсами Базальта — возможно, он там появится после некоторых правок в оформлении.

Охваты. Потенциал рекламных интеграций

Несмотря на «замедление» Ютуба, охваты лекций практически не упали. Среднее количество людей, посещающих первые лекции стабильно высоко, как раньше, так и сейчас. Затем, общее количество просмотров предсказуемо падает. И всё же, речь идёт о десятках, а то иногда и сотнях тысяч просмотров.

Насколько велик потенциал рекламных интеграций в такого рода контенте — вопрос риторический. Здесь нужно решать, а что конкретно можно в таких роликах рекламировать?

Стоит учитывать, что слушатели, в большинстве своём, являются студентами. Например, как вариант, можно размещать рекламу компаний, которым в будущем потребуются специалисты-выпускники. Этот вопрос ещё стоит продумать, но потенциал, несомненно, есть.

Будущее доступности лекций

С сожалением, хотел бы отметить, что в связи с текущей «РКН заварушкой», очевидно, что проводить трансляции на Youtube и выкладывать там ролики без шаманского бубна становится сложно. Альтернативных платформ, подобных Ютубу, к сожалению не было представлено. То, что имеется сейчас, было опробовано и отброшено, как негодный вариант. Скорее всего, если ничего не изменится, ролики будут храниться на некоем диске, откуда их сможет забрать любой желающий. Я, по-возможности, буду всё ещё заливать их на наш канал и туда, куда будет указано договором.

Геннадий Шефель, Сергей Морозов, Даниил Мелешко
Москва, НИИЯФ МГУ

О применения СПО в общем ядерном практикуме Физического факультета МГУ

Аннотация

Хотя студенты ядерного отделения на старших курсах плотно работают с СПО, внедрение СПО в физический практикум происходит медленно из-за отсутствия необходимых драйверов закупаемого оборудования и консервативного отношения сотрудников. Основные цели продвижения СПО: энтузиазм приверженцев и достижение совместимости с современной компьютерной техникой при замене устаревшего оборудования.

Ключевые слова: *практикум по ядерной физике, свободное программное обеспечение, python.*

Современные студенты-ядерщики на старших курсах используют СПО: есть спецкурс по моделированию на Geant4 (взаимодействие

излучения с веществом), проводят обработку результатов используя ROOT, GNU Octave. Под конкретные задачи студенты создают свои программы на Python, C++.

Общий ядерный практикум выполняет порядка 400 студентов на 2-м курсе. Главное при выполнении практических работ по ядерной физике студентами — сущность изучаемых явлений (радиоактивный распад, активация нейтронами...), принципы работы аппаратуры (детектор, АЦП...). Типовая учебная установка задачи ядерного практикума упрощённо: камера с источником радиоактивного излучения и детектор (счётчик Гейгера, полупроводниковый детектор, ФЭУ...). Сигнал с выхода установки оцифровывается с помощью специализированного блока АЦП. Компьютер служит для получения данных через порт ввода/вывода, отображения их в виде спектральных графиков и их математической обработки. В методике обработки ядерно-физического эксперимента превалируют: учёт фона, статистических погрешностей, энергетических потерь; калибровка (соответствие номера канала энергиям частиц). Для реализации основных целей практикума большую часть этих действий студенту следует делать самостоятельно.

В отдельных случаях студенты имеют дело не с реальной установкой, а с моделью эксперимента. Вместо сигнала с реальной установки на вход той же программы задачи приходят с сервера данные, снятые ранее на реальном эксперименте.

Парадоксальность ПО физического практикума в том, что ему, по мнению физиков, следует быть незаметным на фоне изучаемых физических явлений. Для них компьютер — практически «одномерный» инструмент вроде осциллографа с «умным калькулятором» в придачу. Такое отношение к компьютеру сформировало и в целом индифферентное отношение и к железу, и платформе ОС для разработки специализированного ПО.

Сотрудники лаборатории разрабатывали собственные программы для практикума, которые подходят под определение свободного ПО. К примеру, программы обработки экспериментальных кривых на ЭВМ «МИР-1» ещё в 70-х публиковались как приложение в нескольких изданиях описания практикума [1].

В 90–2000-ых гг. АЦП приобретались в ООО ПАРСЕК Дубна (начиная с внутренних плат на ISA, PCI, позднее внешние на USB). Сотрудники практикума создавали программы для измерений и конвертеры форматов данных. Программы предупреждали стандартные

ошибки студентов на установке, в некоторых случаях проводили предварительную обработку данных. В силу отношения к компьютеру как к инструменту программы разрабатывались на базе тех программных средств, что «были под рукой», т.е. на текущей Windows. Драйверы к АЦП предоставлялись производителем только под Windows.

Когда в 2022-2023 гг. началась модернизация практикума, также произошло с поставленными НТЦ «Радэк» блоками ФЭУ с АЦП. Также драйверы только под Windows, и программа, успевшая на текущий момент уже устареть. Перегруженная «профессиональная» программа «Радэка» требовала от студента значительного времени на освоение при том, что, актуальность программы ушла на закате 2007 года. Самодельная программа на Python (PyQt6) оказалась достаточной для удовлетворения большинства требований (задание времени измерения, запись спектра). По мнению одного из авторов, складывается ощущение что современные производители приборов в России не желают считаться с тем что ПО является неотъемлемой частью выпускаемой продукции. Как следствие мы имеем ощутимый вклад в отраслевую отсталость на 20–30 лет. Поэтому любое свободное программное обеспечение отечественного производства должно приветствоваться и поощряться, именно оно будет выступать одной из ступеней фундамента индустриального подъёма.

В 2022–2023 гг. практикуму поставили современные ПК, на которых уже невозможно установить Windows XP или 7 непосредственно. Для запуска устаревших программ задач практикума Linux оказался способом частичного решения проблемы за счёт применения средств виртуализации.

Первым хорошим примером использования СПО в практикуме была установка наглядной регистрации космических лучей «Космический душ» [2] Это мобильная установка, в которой 16 счётчиков Гейгера располагаются над головой на высоте человеческого роста. На крышке расположен одноплатный компьютер Raspberry Pi, блок питания и преобразователь для счётчиков Гейгера. Монитор показывает набор событий в реальном времени на каждой паре счётчиков и суммарный на всех счётчиках. Установка используется на Фестивале Науки, днях Открытых дверей. Помимо показательных выступлений, установка уже несколько лет служит в лаборатории как запасная установка для задачи «Статистика регистрации частиц». Исходный текст программы предоставлен разработчиками и доступен для дальнейшего развития.

В осеннем семестре 2024 года два второкурсника оказались фактически «Евангелистами» применения СПО среди студентов ядерного практикума. Один продемонстрировал обработку полученных данных в SkiDAVis, вместо традиционного Origin, другой построил весьма изящные графики с помощью программы на Python и оставил свои исходные тексты лаборатории.

Литература

- [1] *Практикум по ядерной физике*. Учебное пособие., Изд. 3-е, перераб. и доп., 1979, 190 с.
- [2] Бельшев С. С. *Установка Космический души как одно из средств формирования радиационной грамотности обучающихся во внеурочной деятельности*, // С. С. Бельшев, Е. В. Владимирова, В. В. Вязовский и др. // Наука и школа № 6, 2017

Валерий Баканов

Москва, НИУ ВШЭ

<http://vbakanov.ru/dataflow/>, <http://vbakanov.ru/spf@home/>

Использование метода анализа структуры и целенаправленных преобразований алгоритмов в задачах повышения эффективности параллельных вычислений

Аннотация

В работе приводится опыт применения в процессе обучения студентов метода анализа информационной структуры алгоритмов совместно с целенаправленными эквивалентными её преобразованиями с конечной целью повышения эффективности параллельных вычислений. Преобразования осуществляются с использованием эвристических методов, реализуемых под управлением встроенного скриптового языка. Разработан набор API-вызовов, позволяющий реализовать анализ и преобразования информационных графов конкретных алгоритмов (в частности, их Ярусно-Параллельных Форм — ЯПФ) любой сложности.

Обучающиеся осознают смысл базовых для анализа понятий неограниченного параллелизма, основных параметров ЯПФ и определяющих зависимостей вычислительных практик параллелизма.

Реализация целенаправленных эквивалентных (не изменяющих информационные связи в алгоритмах) преобразований осуществляется путём разработки эвристических методов, при этом используется итерационный подход (пошаговое улучшение качества преобразований в направлении оптимизации заданного параметра). Эвристический подход совместно с итерационным принципом даёт возможность развития исследовательских навыков у обучаемых.

Упор делается на изучение параметров конкретных алгоритмов; предоставляется библиотека часто встречающихся в практике алгоритмов (линейная алгебра, статистика и др.) с возможностью неограниченного дополнения. Данная методика оформлена в виде свободно распространяемой авторской программной системы ПРАКТИКУМ DF-SPF.

Параллельное программирование является одним из наиболее сложных вычислительных практик современности, связано это с необходимостью применения математических методов анализа. В данной работе главенствующим принято направления изучения информационной структуры алгоритмов [1] и исследуются пределы потенциала внутреннего (скрытого) их параллелизма, которые возможно получить в ходе разработки планов параллельного выполнения основанных на данных алгоритмах программ. Приводимые данные базируются на основе практики занятий, проведённых лично автором публикации в период 2018–2023 г. г. при работе со студентами РТУ МИРЭА и НИУ ВШЭ (занятия с бакалаврами, магистрами, научно-исследовательский семинар студентов) и изложенными в книге [2].

Исследовательским инструментом являлась программная система (*учебно-исследовательский инструмент*) ПРАКТИКУМ DF-SPF [3], специально созданный для данных целей. Программные модули разработаны с использованием языка C/C++ в стиле GUI формата PE для модели Win'32, являются полностью OpenSource¹ и могут быть выгружены для свободного использования (формат инсталляционных файлов)². Вычислительные эксперименты проводятся обучаемыми над набором оформленных в виде библиотеки программ (алгоритмов), реализующих наиболее часто применяющиеся на практике алгоритмы (напр., линейной алгебры; библиотека может неограниченно расширяться усилиями участников исследований). Условность выпол-

¹<https://github.com/Valery-Bakanov>

²http://vbakanov.ru/dataflow/content/install_df.exe, http://vbakanov.ru/spf@home/content/install_spf.exe

нения операторов реализуется *предикатным* методом (что позволяет избежать мультивариантности алгоритмов), программные циклы перед выполнением разворачиваются (*unrolling*) с использованием системы макросов.

При этом возможно исследование как параллельных полностью асинхронных систем, так и систем с синхронизацией (напр., в процессорах VLIW-архитектуры с синхронизацией по началу выполнения машинных команд в сверхдлинном слове). Для получения рациональных (разумных, стремящихся к оптимальным вследствие *NP*-полноты задачи [4]) применяются целенаправленные эквивалентные (не нарушающие информационных связей в алгоритме) преобразования ЯПФ информационного графа, осуществляемые с учётом параметров (в частности, формы) ЯПФ. Преобразования задаются с помощью соответствующих API-вызовов скриптового языка Lua [5]. Основной созданием этих сценариев является эвристический подход совместно с итерационным методом движения в сторону повышения качества разрабатываемых планов параллельного выполнения программ.

Характер труда обучаемых предполагает проведение вычислительных экспериментов с использованием разработанных программных средств и на основе полученных данных осмысление базовых свойств, присущих исследованным алгоритмам. Программный модуль ПРАКТИКУМ DF для описания программ использует ассемблероподобную нотацию (мнемоники команд 3-х символьные, порядок операндов соответствует AT&T-синтаксису — «результат правее операндов») в императивном стиле с сохранением принципа единокрatного присваивания. Модуль ПРАКТИКУМ SPF позволяет успешно обрабатывать данные как в условиях микропараллелизма (уровень машинных команд), так и макропараллелизма (значительный размер гранул параллелизма).

Для практической оценки степени использования параллельных вычислительных ресурсов введена характеристика *плотность кода* (один из близких аналогов в области механики — *коэффициент полезного действия*), которую удобно оценивать *коэффициентом использования параллельных вычислителей* (считая общее число таких вычислителей равным ширине ЯПФ после её модификации). Дополнительно анализируется полученная в результате преобразования высота ЯПФ (*параллельная вычислительная сложность*) — оценка времени выполнения алгоритма (программы) и вычислительная сложность собственно преобразований (в единицах, аналогичных использован-

ным в традиционных методах анализа сложности сортировки массивов). Важно, что полученная на этапе анализа именно алгоритмов (естественно, посредством анализа их ЯПФ) величина потенциала параллелизма является предельной (что важно для оценки возможной степени параллелизации на программном уровне).

Таким образом выявляется триединая направленность данной работы:

1. Слушатели глубже понимают роль **алгоритма** при разработке реальных программ и знакомятся со *стандартными алгоритмами* обработки данных.
2. Осознают **потенциал внутреннего параллелизма** как неотъемлемую часть (грань) сущности **алгоритм** (природу понятия, его основные параметры, единицы измерения, специфику применения).
3. Естественным образом воспринимают принципиальную возможность целенаправленных эквивалентных преобразований алгоритмов с целью наилучшего использования их в параллельных вычислительных практиках (в смысле приспособленности к конкретным вычислительным системам) и разрабатывают эффективные процедуры таких преобразований (*творческий компонент* работы).

Программный ПРАКТИКУМ DF-SPF применяется при исследованиях в Университетах России, показал себя действенным учебно-исследовательским инструментом и разработчик заинтересован в дальнейшем её распространении.

Литература

- [1] Воеводин В. В. *Вычислительная математика и структура алгоритмов* (учебник, 10 лекций). // — М., Изд. МГУ, 2010. — 168 с.
- [2] Баканов В. М. *Практический анализ алгоритмов и эффективность параллельных вычислений*. // — М.: Пробел-2000, 2023. — 198 с. (<https://clck.ru/39sdgk>).
- [3] V. M. Bakanov. *Software complex for modeling and optimization of program implementation on parallel calculation systems*. // Open Computer Science, 2018, 8, Issue 1, Pages 228–234, DOI: <https://www.degruyter.com/document/doi/10.1515/comp-2018-0019>

- [4] М. Гэри, Д. Джонсон. *Вычислительные машины и труднорешаемые задачи.* // — М.: Мир, 1982. — 416 с.
- [5] Иерузалымски Роберту. *Программирование на языке Lua.* // — М.: ДМК Пресс, 2014. — 382 с.

Евгений Алексеев, Юлия Чуракова, Давид Головатин, Ирина Горбунова

Краснодар, Кубанский государственный университет,
math.kubsu.ru

Использование языка программирования Julia на факультете математики и компьютерных наук Кубанского Государственного Университета

Аннотация

Описан опыт использования языка программирования Julia на факультете математики и информатики Кубанского Государственного Университета. Представлено учебное пособие по языку Julia в формате Jupyter.

Ключевые слова: *Julia, Jupyter, Система Открытой Математики.*

Язык программирования Julia¹ распространяется под свободной лицензией MIT². К особенностям языка следует отнести:

- JIT-компиляция и, как следствие, высокая скорость работы;
- синтаксис языка, схожий с Scilab, MATLAB;
- большое количество пакетов для решения различных задач;
- наличие REPL среды, поддержка языка Julia в Geany, Jupyter.

В Кубанском Государственном Университете Julia широко использует профессор Рожков А.В. вместе со своими учениками в научных исследованиях [1, 2].

Авторами была поставлена задача познакомить широкий круг студентов факультета с этим языком, чтобы они могли использовать его в своих исследованиях. В 2022–23 учебном году студентам выдавались

¹<https://julialang.org/>

² <https://mit-license.org/>

индивидуальные задания по освоению языка и решению математических задач. В 2023–24 учебном году были прочитаны обзорные лекции по языку в рамках курсов «Технологии программирования и работа на ЭВМ», «Программирование», «Математические пакеты и их применение в естественно-научном образовании». Кроме того на факультете начало развиваться направление, посвящённое генерации специализированных математических пакетов. В рамках этого направления разрабатывается Система Открытой Математики (СОМ) [3, 4]. Изначально СОМ строилась на базе Julia и Jupyter, сейчас туда в качестве вычислительных ядер добавились Scilab и Python. Параллельно с этими исследованиями в 2024–25 учебном году одним из авторов в рамках дополнительного образования в КубГУ был прочитан курс по математическому моделированию на базе языка Julia. Далее планируется полноценное внедрение языка Julia наряду с Scilab в курс «Математические пакеты и их применение в естественно-научном образовании».

В свете выше изложенного возникла необходимость разработки учебного пособия по языку программирования Julia. Авторами было принято решение не только описать возможности языка, но и рассмотреть его использование при решении различных математических задач (обработка эксперимента, несложные задачи математического моделирования, задачи теории чисел и криптографии). Учебное пособие разрабатывается в формате Jupyter и является интерактивным. На ФМиКН КубГУ накоплен достаточный опыт использования Jupyter в учебном процессе [5, 6]. В состав учебного пособия [7] входят следующие разделы:

1. *Общие сведения о языке Julia.* Вводный раздел пособия посвящён основам языка Julia. Первые параграфы посвящены установке Julia и Jupyter на компьютер пользователя. Далее внимание уделяется базовым концепциям, таким как переменные и их адреса, их присваивание и использование, с особым акцентом на систему типов данных. Julia сочетает в себе элементы как статической, так и динамической типизации, что позволяет разработчикам гибко подходить к определению типов переменных и структур данных. В разделе также подробно объясняются встроенные функции языка Julia, которые позволяют эффективно решать математические и программные задачи. Этот раздел

создаёт прочную основу для дальнейшего изучения языка Julia, его возможностей и применения в решении реальных задач.

2. *Поддерживаемые структуры данных (строки, массивы, кортежи, словари).* Рассматриваются основные структуры данных, поддерживаемые языком программирования Julia, такие как строки, массивы, кортежи и словари. Подробно объясняется, как создавать и использовать структуры данных, а также какие операции можно выполнять с ними, что является основой для дальнейшей работы с Julia и решения более сложных задач.
3. *Управляющие конструкции языка Julia.* Авторы описывают управляющие конструкции языка Julia, такие как условные операторы (if, elseif, else) и циклы (for, while).
4. *Функции в Julia.* Раздел посвящён функциям в языке Julia, которые являются основным инструментом для организации кода в модульные блоки, позволяющие повторно использовать и структурировать программу. Рассматриваются способы определения и вызова функций, а также особенности передачи аргументов и возврата значений.
5. *Примеры программ на Julia.* Представлены конкретные программы, написанные на языке Julia. Демонстрируются различные аспекты работы с языком, включая использование базовых структур данных, циклов, условий и функций. Приведённые коды помогают понять, как применять синтаксис и возможности языка Julia для решения математических задач.
6. *Пакеты в Julia.* Обучающийся знакомится с пакетами в языке Julia, которые предназначены для расширения функциональности языка. Пакеты позволяют использовать уже готовые решения для часто встречающихся задач, таких как анализ данных, визуализация и численные вычисления. В разделе рассматриваются основные механизмы работы с пакетами, включая их установку и загрузку.
7. *Визуализация данных в Julia (пакет Plots).* Рассказывается о визуализации данных с использованием пакета Plots в Julia. Рассматриваются основные возможности для создания и форматирования графиков и диаграмм.
8. *Решение задач линейной алгебры.* Рассматриваются средства решения задач линейной алгебры в языке Julia. Julia обладает

мощной поддержкой линейной алгебры, предоставляя оптимизированные функции для выполнения операций с матрицами, таких как умножение, транспонирование, вычисление обратных матриц и решение систем линейных уравнений.

9. *Файлы в Julia*. Авторы рассказывают об особенностях работы с текстовыми и двоичными файлами.
10. *Решение задач обработки эксперимента*. На практических примерах рассматривается решение задач аппроксимации и интерполяции с помощью языка программирования Julia.
11. *Решение задач математического моделирования*. Обучающийся знакомится с решением задач математического моделирования в языке Julia, включая работу с обыкновенными дифференциальными уравнениями (ОДУ) и дифференциальными уравнениями в частных производных (УЧП). Рассматриваются различные методы численного решения дифференциальных уравнений, такие как методы Эйлера, Рунге-Кутты и другие, а также использование специализированных библиотек.
12. *Задачи теории чисел*. Раздел посвящён задачам теории чисел, которые могут быть решены с использованием языка Julia. Рассматриваются различные алгоритмы для работы с простыми числами, разложением на простые множители, нахождением наибольшего общего делителя (НОД) и другие теоретико-числовые задачи. Язык предоставляет хорошие возможности для разработки эффективных алгоритмов для теории чисел.
13. *Задачи криптографии*. Рассматриваются задачи криптографии, которые можно решать с использованием языка Julia. Рассматриваются алгоритмы шифрования и дешифрования, такие как RSA. Эти технологии важны для разработки безопасных информационных систем и защиты данных.

Формат Jupyter является довольно удачным для учебного пособия по языку Julia. Обучающиеся изучают материал и могут запустить любую программу из пособия, изменить её и посмотреть на результат.

Литература

- [1] Рожков, А. В. *Экспериментальная математика на платформе языка программирования Julia* / А. В. Рожков // Материалы Всероссийской

- научно-практической конференции «Тенденции и перспективы развития обучения математике и информатике в условиях реальной и цифровой среды», Краснодар, 28 марта 2023 года. — Краснодар: Кубанский государственный университет, 2023. — С. 30–36. — EDN TISLQI.
- [2] Рожков, А. В. *Язык программирования Julia — современное средство обучения математике* / А. В. Рожков, И. Л. Ойнас, М. В. Цалюк // Математика и математическое образование в эпоху цифровизации : материалы XII Всероссийской с международным участием научно-методической конференции, Красноярск, 09–10 ноября 2023 года. — Красноярск: Красноярский государственный педагогический университет им. В.П. Астафьева, 2023. — С. 117–122. — EDN ZMRYWK.
- [3] Github — EnckyOff/SysOpenMath. URL: <https://github.com/Encky0ff/SysOpenMath> (дата обращения: 3.01.2025).
- [4] Головатин Д. Т. *Инструменты разработки специализированных математических пакетов* / Д. Т. Головатин, Е. Р. Алексеев, Е. А. Вербичева, К. В. Дога, Д. Е. Юрченко. // Системы компьютерной математики и их приложения: межвузовский сборник научных трудов. Смоленск: Изд-во СмолГУ, 2024. Вып. 25. — С.15–20.
- [5] Алексеев Е. Р. *Scilab: Решение инженерных и математических задач: учеб. издание* / Е. Р. Алексеев, К. В. Дога, О. В. Чеснокова. / отв. ред. В. Л. Чёрный. — М.: Базальт СПО; ДМК Пресс, 2024. — 440 с.
- [6] Алексеев Е. Р. *Jupyter как универсальная среда для обучения и научных исследований* / Е. Р. Алексеев, Е. А. Вербичева, К. В. Дога, Д. Т. Головатин, Д. Е. Юрченко. // Системы компьютерной математики и их приложения: межвузовский сборник научных трудов. Смоленск: Изд-во СмолГУ, 2024. Вып. 25. — С.3–9.
- [7] Учебное пособие «Использование Julia при решении математических задач» URL: <https://github.com/JuliaChurakova/Julia> (дата обращения: 3.01.2025).

Елена Татьянич, Павел Жданович

Волгоград, ФБГОУ ВО «Волгоградский государственный социально-педагогический университет»

Проект: Курс «3D-моделирование и печать»

http://lms.vspu.ru/courses/3d-modelling_and_3d-printing/

Обучение 3D-печати в виртуальной среде

Аннотация

Представлена виртуальная машина под управлением Simply Linux со свободным ПО, которая является средой выполнения практических заданий авторского курса 3D-печати. Машина доступна для скачивания. Преподаватель может подключиться к ней он-лайн, используя распределённое образовательное облако МИФ ВГСПУ.

Ключевые слова: *Simply Linux, Blender, Cura, FreeCad, KVM, VirtualBox.*

С 2020 года на факультете математики, информатики и физики Волгоградского государственного социально-педагогического университета ведётся авторский курс «3D-моделирование и печать» для бакалавров направления «Педагогическое образование» профилей «Математика» и «Информатика», «Информатика» и «Технология».

В последнее время растёт число студентов, осваивающих программу обучения по индивидуальным планам или желающих изучать дисциплину дистанционно. Для таких студентов нами создана виртуальная машина, реализующая полную и минимальную среду выполнения лабораторных работ этого курса на свободном ПО. Копии машины запускаются на очных занятиях в компьютерных аудиториях или на личных компьютерах студентов.

Операционная система Simply Linux 10.4. В её составе есть всё необходимое вспомогательное ПО: офисный пакет LibreOffice, браузер Chromium, архиватор, просмотрщик pdf-файлов. Из репозитория установлено необходимое для реализации курса ПО: для создания и начальной подготовки к печати трёхмерных моделей — пакет 3D-графики Blender и САПР FreeCAD, для окончательной подготовки к печати — слайсер Ultimaker Cura [1].

Для дистанционной работы студенты скачивают виртуальную машину в виде образа для Oracle VirtualBox или KVM из облачного хранилища NextCloud, развёрнутого на факультете. Ссылка для скачивания приведена на главной странице курса.

Для запуска виртуальной машины рекомендуются следующие минимальные параметры:

1. основная память: 4096 МБ;
2. процессор: 2 ядра;
3. предел загрузки ЦПУ: 100%;
4. видеопамять: 64 МБ;
5. графический контроллер: VMSVGA;
6. 3D-ускорение можно отключить.

Задания лабораторных работ доступны на портале электронного обучения ВГСПУ по адресу http://lms.vspu.ru/courses/3d-modelling_and_3d-printing/, в том числе из виртуальной машины. Обучающиеся отправляют выполненные работы на проверку на соответствующие страницы курса.

Преподаватель в целях контроля может получить доступ к студенческой виртуальной машине дистанционно, используя службу удалённого доступа к рабочему столу или по SSH. Для этого копия виртуальной машины автоматически подключается к факультетскому серверу VPN, используя личный сертификат, заранее выданный студенту. Таким образом возникает распределённое образовательное облако, в котором граничными ресурсами являются виртуальные машины студентов.

Размещение виртуальных ресурсов на компьютерах обучающихся является оптимальным решением, так как полноценная работа с 3D-графикой и САД-системами через Интернет ограничена скоростью каналов и ресурсами гипервизоров.

Виртуализация позволяет обеспечить:

1. дистанционность выполнения лабораторных работ под контролем преподавателя;
2. единообразие версий используемого студентами ПО, что обеспечивает отсутствие ошибок открытия преподавателем файлов выполненных работ, критически важное при использовании сред создания трёхмерных моделей;
3. соответствие версий ПО, рассматриваемых в текстах лабораторных работах, версиям ПО, используемым на практике.

Авторы приглашают коллег воспользоваться разработанным курсом 3D-печати и распределённым образовательным облаком МИФ ВГСПУ для обучения своих студентов.

Литература

- [1] Татьянач, Е. *Использование свободного программного обеспечения при подготовке будущих учителей информатики в области 3D-печати* / Е. Татьянач // Девятнадцатая конференция «Свободное программное обеспечение в высшей школе» : Материалы конференции, Переславль-Залесский, 28–30 июня 2024 года. — Москва: ООО «МАКС Пресс», 2024. — С. 51–54. — EDN YIJZZS.

Екатерина Прокофьева

Москва, Национальный исследовательский университет «Высшая школа экономики», департамент компьютерной инженерии, Московский институт электроники и математики имени А. Н. Тихонова

<https://cabinet.miem.hse.ru/project/2142/0/passport>

Разработка практико-ориентированных заданий в обучении UML-программирования

Аннотация

Диаграммы UML применяют в проектировании, презентациях, описании или создании документации, в данном проекте мы хотим отработать автоматизацию процессов выстраивания визуальных моделей по процессам программирования. Современные инструменты позволяют рассмотреть систему со всех точек зрения, имеющих отношение к её разработке и последующему развёртыванию технических работ по организации сетей, инженерным расчётам, технико-экономическому контролю. Разработка ориентированных на реальный сектор практических заданий и интерактивных мультимедийных презентационных материалов, посвящённых моделированию программного обеспечения. Материалы будут включать в себя подробные инструкции, примеры, кейсы, а также визуальные презентации. Основное внимание уделено обучению основным методам построения диаграмм классов, последовательностей, состояний и других UML-диаграмм, а также применению этих инструментов для проектирования и анализа программных систем для ИТ-компаний, логистики и промышленности etc. Важным аспектом становится использование инструментов создания UML-диаграмм на основе свободного ПО и разработки собственных программных приложений/решений. Сформированные задания будут способствовать формированию практических навыков моделирования реальных проектов

и углублению понимания принципов проектирования ПО, что актуально и востребовано на ИТ-рынке труда.

Также Linux-подобные ОС — прекрасная платформа для реализации автоматизированного инструментария.

Ключевые слова: *UML-модели, ключевые показатели развития, подсистемы отчётности.*

Введение

В условиях реализации стратегии импортозамещения показать обучающимся возможности и преимущества отечественных операционных систем на примере ALT Linux в решении актуальных прикладных инженерных, статистических и исследовательских задач: запуск процессов в рамках базового функционала, разграничения прав, работы с репозиториями, установки библиотек и пакетов, функционирования образовательных, программных, статистических, графических, сетевых, ГИС и специализированных управленческих приложений.

Установить автоматизированные, графические и полуавтоматизированные инструменты в рамках Альт Образование 10.4.

Актуальные тематические направления: Основы работа с большими данными в различных областях. Основы программирования и работы с кодом на базе пакетов и приложений. Работа с пространственными данными. Работа с социально-экономическими задачами. Работа с инженерно-техническими и научными задачами. Работа с графикой и видео. Работа с технической документацией. Работа с сетью и интернет-приложениями. Работа с нефинансовой и экологической отчётностью. Обеспечение безопасности данных.

Примеры решений

Draw.io и все подобные инструменты — это онлайн-сервисы, которые помогают быстро создавать и редактировать различные блок-схемы, диаграммы и другие объекты, которые часто связаны с проектной деятельностью, описанием организационных регламентов и структур, распределением ролей и задач. Таких инструментов много, среди них большую популярность завоевали PlantUML, Mermaid и, собственно, draw.io.

Приложения для ручной отрисовки UML-диаграмм Возможности:

- Редактирование вручную: Пользователь сам выбирает тип диаграммы (например, диаграммы классов, прецедентов, последовательностей) и размещает элементы на холсте.
- Богатый набор инструментов: Большинство таких приложений предоставляют набор предустановленных фигур (классы, интерфейсы, соединения и т.д.).
- Настройка стилей: Поддержка настройки шрифтов, цветов, толщины линий и других визуальных параметров.
- Совместимость: Возможность экспорта в популярные форматы (PDF, PNG, SVG) и интеграции с другими инструментами (например, Confluence, Jira).
- Поддержка совместной работы: Некоторые приложения (например, Lucidchart или Creately) позволяют редактировать диаграммы нескольким пользователям в реальном времени.

Примеры ПО:

- Microsoft Visio
- Lucidchart
- Draw.io (или Diagrams.net)
- StarUML

Преимущества:

- Высокая гибкость в дизайне: диаграммы можно настраивать до мельчайших деталей.
- Удобство для небольших проектов или для пользователей без навыков программирования.
- Быстрое прототипирование: легко набросать схему для обсуждения с командой.

Недостатки:

- Затратно по времени для сложных или больших диаграмм.
- Риск ошибок из-за отсутствия проверки синтаксиса или логики.
- Низкая автоматизация: изменения в модели требуют ручного обновления диаграммы.

Программное обеспечение для автоматического построения UML-диаграмм

Возможности:

- Генерация по коду: Программы могут анализировать исходный код (например, Java, C#) и автоматически строить диаграммы классов, последовательностей и т.д.
- Интеграция с IDE: Генерация диаграмм из среды разработки (например, IntelliJ IDEA, Eclipse).
- Анализ баз данных: Построение диаграмм на основе схемы базы данных (ER-диаграмм).
- Обратное проектирование (reverse engineering): Генерация диаграмм из существующего ПО.
- Синхронизация: Автоматическое обновление диаграмм при изменении кода.

Примеры ПО:

- PlantUML
- Enterprise Architect
- Visual Paradigm
- Umbrello

Преимущества:

- Быстрота: автоматическое создание диаграмм экономит время.
- Снижение риска ошибок: диаграммы строятся на основе кода, что минимизирует расхождения между проектом и диаграммами.
- Подходит для крупных проектов: особенно полезно, если нужно визуализировать обширные системы.

Недостатки:

- Ограниченная кастомизация: автоматические диаграммы могут быть сложны для настройки.
- Требуется знаний о синтаксисе языка моделирования или использования ПО.
- Может быть сложным для начинающих из-за необходимости настройки.

Построение UML-диаграмм программно в неспециализированных приложениях.

Возможности:

- Написание скриптов: создание диаграмм через код с использованием специальных библиотек (например, PlantUML, Graphviz).
- Гибкость интеграции: поддержка интеграции с CI/CD, генерации диаграмм в документацию.
- Масштабируемость: можно автоматизировать построение диаграмм для сложных систем или многократно используемых шаблонов.
- Взаимодействие с другими инструментами: встраивание диаграмм в документы, отчёты или веб-приложения.

Примеры инструментов:

- PlantUML
- Python-библиотеки (например, diagrams)
- Mermaid.js (для веб-интеграций)

Преимущества:

- Высокая автоматизация: диаграммы генерируются из описаний, что минимизирует ручной труд.
- Динамическое обновление: изменения в коде автоматически отражаются на диаграммах.
- Удобство работы в команде: можно интегрировать диаграммы в процессы разработки.

Недостатки:

- Необходимость программных навыков.
- Более низкая визуальная кастомизация, чем в специализированных инструментах.
- Могут быть сложности в отладке больших диаграмм.

Заключение

Некоторые недостатки работы с PlantUML:

- Сложности с обучением. Синтаксис простой, но пользователям, незнакомым с текстовым созданием диаграмм, может потребоваться время на освоение. Отсутствие графического интерфейса. Это может быть недостатком для тех, кто предпочитает создание диаграмм перетаскиванием элементов.

- Трудоемкость работы со сложными диаграммами. Для очень сложных диаграмм текстовый синтаксис может быть громоздким и сложным в управлении. Ограничения рендеринга. Параметры стиля и форматирования менее гибкие, чем в некоторых специализированных графических инструментах для создания диаграмм.
- Низкая производительность. Для больших диаграмм процесс рендеринга текста в диаграмму может быть медленным.
- Проблемы с безопасностью. Использование PlantUML с опциями удаленного сервера может вызывать проблемы с безопасностью, особенно при работе с чувствительной информацией.

Некоторые недостатки работы с Mermaid:

- Ограничения по сложности диаграмм. Для очень сложных диаграмм Mermaid может не обеспечивать необходимую детализацию и гибкость.
- Проблемы с рендерингом. Могут возникать несоответствия рендеринга на разных платформах и в браузерах, что влияет на внешний вид и надёжность диаграмм.
- Необходимость глубокого понимания синтаксиса и структуры. Для создания сложных диаграмм может потребоваться более глубокое понимание синтаксиса и структуры.
- Ограниченная настройка. Mermaid может не предлагать уровень настройки, который доступен в некоторых специализированных инструментах для создания диаграмм, ограничивая эстетические и функциональные модификации.
- Зависимость от текстового ввода. Это может быть недостатком для тех, кто предпочитает интерфейсы перетаскивания элементов или визуальное редактирование на основе графического интерфейса.

Выбор между PlantUML и Mermaid зависит от конкретных требований и предпочтений пользователя.

Ручная отрисовка удобна для небольших проектов или презентаций, когда нужна высокая визуальная детализация.

- Автоматическое построение подходит для крупных проектов, где важна скорость и точность, а диаграммы нужны для анализа существующего кода.

- Программное построение лучше всего для автоматизации и интеграции в рабочие процессы, особенно в контексте DevOps и CI/CD.

Оба ключевых программных инструмента UML-решений работают на базе Chromium и дополняются пакетами репозитория на базе Альт Образование.

Литература

- [1] Ларман, К. *Применение UML 2.0 и шаблонов проектирования* : Введение в объектно-ориентированный анализ, проектирование и итеративную разработку : пер. с англ., К. Ларман, 978-5-907144-36-12019
- [2] Галиаскаров, Э. Г. *Анализ и проектирование систем с использованием UML* : учебное пособие для вузов / Э. Г. Галиаскаров, А. С. Воробьев. — Москва : Издательство Юрайт, 2023. — 125 с. — (Высшее образование). — ISBN 978-5-534-14903-6. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/520341> (дата обращения: 16.01.2025)
- [3] UML Web Site — <https://www.uml.org/>

Тамара Зайка, Денис Зайка

Донецк, ФГБОУ ВО «Донецкий государственный медицинский университет имени М. Горького» МЗ РФ

Проект: OpenDocument-шаблон для простого макета рукописи российской кандидатской диссертации <https://github.com/ventricola/Russian-Phd-OpenDocument-Dissertation-Template>

Простота и сложность подготовки рукописи кандидатской диссертации в Libreoffice

Аннотация

Представлены преимущества и недостатки применения Libreoffice Writer для подготовки простой рукописи кандидатской диссертации, соответствующей нормативам, создан и опубликован в открытом репозитории шаблон для этого документа. Даны некоторые практические рекомендации для подготовки макета такой рукописи.

Ключевые слова: *Libreoffice, шаблон, кандидатская диссертация, ГОСТ Р 7.0.11-2011.*

Подготовка текста рукописи кандидатской диссертации в России регламентируется ГОСТ Р 7.0.11-2011, который, в свою очередь, ссылается ещё на ряд ГОСТов [3]. Поэтому несоответствующий данным требованиям документ может быть не принят и отправлен на доработку экспертными советами, что может вызвать сложности с защитой диссертации.

Существует несколько свободных программных инструментов, часто использующихся для создания макета диссертации, наиболее популярными из которых являются различные клоны текстового процессора OpenOffice Writer (Libreoffice Writer, в частности), а также процессоры TeX (например, LaTeX [1]). Также существуют способы использовать другие языки разметки (например, markdown с pandoc [4] и т.п.).

При подготовке собственной работы медико-биологического направления автором был выбран Libreoffice Writer, макет рукописи работы достаточно прост, в нем нет объёмных формул, но есть некоторое количество таблиц и иллюстраций. TeX требует серьёзной подготовки для использования, а языки разметки не позволяют сверстать документ с достаточной точностью, соответствующий всем требованиям. При этом Writer лёгок в освоении и использовании, функционален и широко применяется.

В первую очередь в начале работы был создан шаблон [2], в котором определены стили и параметры документа, соответствующие нормативам, для основного текста, заголовков разного уровня, таблиц и рисунков, оглавления и библиографии, колонтитулов первой страницы, чётных и нечётных страниц. В качестве элементов библиографических ссылок, а также ссылок на таблицы и рисунки решено было использовать поля-переменные «Диапазон нумерации», а далее поля-перекрёстные ссылки на элементы диапазонов. Этот способ значительно проще использования встроеного во Writer стандартного инструмента библиографии. При этом библиографические источники можно легко редактировать прямо в тексте, добавлять, удалять и сортировать элементы библиографии, а ссылки автоматически обновляются. Ссылки на удалённые элементы легко искать. Для вставки рисунков используются врезки, все графические элементы вставляются с привязкой «Как символ», что предотвращает расползание текста.

В целом, функциональности Libreoffice Writer достаточно для подготовки не очень сложного структурированного документа, а освоить его значительно проще, чем LaTeX.

К сожалению, в процессе подготовки документа обнаружились некоторые сложности.

Поиск даже в последних версиях Writer плохо ищет в тексте полей, а заменой сложно пользоваться для редактирования специальных символов, поэтому автор использовал дополнение AltSearch, которое значительно функциональнее встроенного инструмента.

В процессе работы обнаружилось, что некоторые коллеги работают только с документами Microsoft Word, а при сохранении или конвертации документа Writer или открытии .odt в пакете Microsoft часть разметки искажается, что требует потом ручной корректировки. К счастью, многие готовы работать с PDF, а сохранение в этот формат уже много лет работает безотказно.

Литература

- [1] LaTeX-шаблон для русской кандидатской диссертации и её автореферата, [Электронный ресурс] URL: <https://github.com/AndreyAkinshin/Russian-Phd-LaTeX-Dissertation-Template> (дата обращения: 05.01.2025).
- [2] OpenDocument-шаблон для простого макета рукописи российской кандидатской диссертации, [Электронный ресурс] URL: <https://github.com/ventricola/Russian-Phd-OpenDocument-Dissertation-Template> (дата обращения: 09.01.2025).
- [3] ГОСТ Р 7.0.11—2011, Система стандартов по информации, библиотечному и издательскому делу. Диссертация и автореферат диссертации. Структура и правила оформления., 2011
- [4] Зенкин В. А., *Применение языка разметки Markdown для написания отчётов о НИР и других сложноструктурированных документов с регламентированными требованиями к оформлению* // Восемнадцатая конференция Свободное программное обеспечение в высшей школе : материалы конференции. Переславль-Залесский, 2023. С. 73–76.

Евгений Алексеев, Ирина Горбунова, Юлия Чуракова
Краснодар, Кубанский государственный университет,
math.kubsu.ru

Среда изучения команд терминала Linux

Аннотация

Представлена разработанная авторами среда для изучения команд терминала. Описан опыт её использования на факультете математики и компьютерных наук Кубанского Государственного Университета.

Ключевые слова: *Linux, терминал, Jupyter, система обучения.*

Освоение свободного программного обеспечения зачастую начинается со знакомства с операционной системой семейства Linux. При освоении Linux пользователь в первую очередь должен изучить файловую систему и команды терминала Linux. Однако, современным пользователям зачастую сложно адаптироваться к командам терминала, сказывается привычка работать в графическом режиме. Это создаёт необходимость разработки эффективных методов обучения, которые помогут преодолеть барьер между графическим интерфейсом и командной строкой. Для эффективного обучения командам терминала требуется:

1. Иметь учебник и методическое пособие.
2. На компьютер обучающегося предварительно установить операционную систему семейства Linux.
3. Иметь индивидуальный набор заданий по каждой теме и варианты контрольных работ для каждого ученика.

О решении создании учебника и методического пособия мы рассказывали на объединённой конференции в Переславле в 2022 году [1]. Данная работа описывает разработанную авторами среду для изучения терминала Linux (СИТЛ)¹ на базе Jupyter [2]. Среда состоит из модифицированного Jupyter Notebook (в Jupyter устанавливается ядро `bash — pip install bash_kernel`) и скрипта, создающего систему файлов и каталогов для выполнения заданий и контрольных работ.

Скрипт генерирует индивидуальные варианты заданий для каждого обучающегося, случайным образом выбирая элементы из заранее подготовленных файлов, и обеспечивает их защиту от изменений.

¹СИТЛ — система изучения терминала Линукс.

Преподаватель имеет возможность редактировать эти файлы, прилагающиеся к скрипту. Кроме того, скрипт, используя свободное приложение graphviz [3], создаёт уникальное изображение дерева каталогов (рис. 1) для одного из заданий.

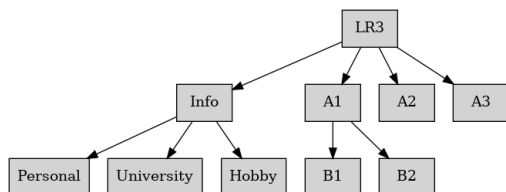


Рис. 1: Файловая структура индивидуального задания.

Файл для обучающегося включает в себя инструкцию для работы и текстовые ячейки с заданиями, после которых предусмотрены ячейки для их выполнения (рис. 2).

Индивидуальная учебная среда упрощает выполнение заданий и ускоряет процесс проверки работ.

Разработанная авторами среда может быть адаптирована под конкретные задачи обучения, что делает её универсальным инструментом.

СИТЛ на завершающем этапе может (опционно) генерировать итоговое зачётное задание.

Среда начала внедряться в курсы «Программное обеспечение ЭВМ», «Использование свободных и отечественных операционных систем» на факультете математики и компьютерных наук КубГУ в процессе обучения. Также начато использование СИТЛ для самостоятельного освоения студентами операционных систем семейства Linux в различных дисциплинах информационного профиля.

Литература

- [1] Алексеев, Е. Р. Учебное пособие «Первое знакомство с ОС Linux» / Е. Р. Алексеев, К. В. Дога, Ю. Н. Номоконова // Объединённая конференция «СПО: от обучения до разработки»: Сборник тезисов конференции. Переславль-Залесский, 19–22 мая 2022 года. / отв. ред. В. Л. Чёрный. — Москва: ООО «МАКС Пресс», 2022. — С. 53–56. — EDN FDDCWV.

Jupyter LR Last checkpoint: 23 days ago

File Edit View Run Kernel Settings Help Trusted

JupyterLab Bash

Лабораторная работа №3

Задания данной лабораторной работы выполняются в каталоге LR3. Запустите ячейку ниже, чтобы отобразить рисунок.

1. В каталоге создать дерево каталогов как показано на рисунке (файл LR3/task.dot.png).
2. В каталоге A2 создать подкаталоги B4 и B5 и удалить каталог B1.
3. Используя терминал, в каталоге Personal создать файл name.txt, содержащий информацию о фамилии, имени и отчестве студента. Здесь же создать файл Date.txt, содержащий информацию о дате рождения студента. В этом же каталоге создать файл school.txt, содержащий информацию о школе, которую закончил студент. Сохранить скриншоты выполнения задания в каталоге LR3.
4. Используя терминал, в каталоге University создать файл name.txt, содержащий информацию о названии Вуза и специальности, на которой студент обучается. Сохранить скриншоты выполнения задания в каталоге LR3.
5. Используя терминал в каталоге Hobby создать файл hobby.txt с информацией об увлечениях студента. Сохранить скриншоты выполнения задания в каталоге LR3.
6. Скопировать файл hobby.txt в каталог A2.
7. Объедините файлы hobby.txt, name.txt, school.txt в файле all.txt каталога Info.
8. Удалите файлы hobby.txt, name.txt, school.txt.
9. Выведите содержимое файла all.txt на экран.
10. Скопировать файл all.txt в директорию A1.

```

graph TD
    LR3 --> Info
    LR3 --> A1
    LR3 --> A2
    Info --> Personal
    Info --> University
    Info --> Hobby
    A1 --> B1
    A1 --> B2
  
```

```

[12]: cd ../LR3
      mkdir Info A1 A2 A3
      cd Info
      mkdir Personal University Hobby
      cd ../A1
  
```

Рис. 2: Окно СИТЛ с заданием

- [2] СИТЛ — система изучения терминала линукс URL: https://github.com/UserName-IrinaGR/Jup_Lin (дата обращения: 5.01.2025).
- [3] Graphviz URL: <https://graphviz.gitlab.io/> (дата обращения: 5.01.2025).

Дмитрий Пилюк

Санкт-Петербург, Embox

<https://github.com/embox/embox>

Использование СПО в сфере АСУ ТП на примере ОС RV Embox

Аннотация

В работе рассматривается СПО в сфере АСУ ТП, описывается процесс расширения возможностей ОС RV Embox и использования с другими инструментами для разработки и запуска ПО в данной области.

Ключевые слова: *АСУ ТП, ОС RV Embox.*

В настоящее время наблюдается развитие и повсеместное применение технологий автоматизации. Одной из значимых областей автоматизации является управление технологическими процессами. В сложных промышленных системах с большим количеством взаимосвязанных элементов эффективность работы напрямую зависит от скорости обработки информации и точности принимаемых решений. Здесь на первый план выходят автоматизированные системы управления технологическими процессами (АСУ ТП).

Основной задачей в данной области является программирование ПЛК и организация их взаимодействия между собой, что описано в стандарте IEC 61131.

IEC 61131 — набор стандартов МЭК для программируемых контроллеров.

IEC 61131-3 [3] — раздел международного стандарта IEC 61131, описывающий языки программирования для программируемых логических контроллеров. Включает языки LD, FBD, SFC, ST, IL.

Существующие решения

В данной работе были рассмотрены и использованы следующие инструменты:

- **Matiec** [4] — компилятор с открытым исходным кодом для языков программирования, определённых в стандарте IEC 61131-3. Проект предоставляет 2 исполняемых файла `iec2c` и `iec2iec`. В данной работе используется `iec2c`, программа транслирующая код на языке **ST** в код на языке **C**.

- **Beremiz** [1] — это интегрированная среда разработки для автоматизации машин. Это свободное программное обеспечение, соответствующее, среди прочих стандартов, стандарту IEC-61131. Так же предоставляет свою среду выполнения для программ. Поддерживает различные протоколы взаимодействия. Для генерации программ использует **Matiec**.

Эти свободные инструменты позволяют разрабатывать прикладные программы языков IEC 61131-3 (МЭК 61131-3).

Процесс следующий, в IDE разрабатывается программа на графическом языке и транслируется в язык ST. Затем **Matiec** компилирует этот файл в **.c** и **.h**, а с ними уже работает **GCC** или **Clang** и программа готова к запуску на устройстве.

Modbus

В сфере АСУ ТП необходимо, чтобы устройства передавали информацию между собой. Это взаимодействие описывают различные протоколы. Одним из таких является **Modbus** — открытый коммуникационный протокол, основанный на архитектуре ведущий — ведомый (англ. master-slave).

Embox уже поддерживает реализацию данного протокола библиотекой **Libmodbus**¹.

АСУ ТП на ОС PV Embox

Цель работы: предоставить возможность использовать **Embox** в качестве целевой операционной системы.

Для этого было сделано:

1. Модифицирован процесс сборки в **Beremiz**².
2. Добавлена генерация header файла для программ, использующих **Modbus**.
3. Добавлена цель сборки **.st** файлов в **Mybuild**³.
4. Реализована среда выполнения для файлов, полученных в результате работы **Matiec**.

¹<https://github.com/stephane/libmodbus>.

²Был сделан форк репозитория **Beremiz** https://github.com/embox/beremiz/tree/add_embox_target.

³Система сборки используемая в **Embox**.

Эксперимент

Для проверки результата работа было решено написать небольшую программу, которая управляет светодиодом по Modbus. Для этого в Beremiz необходимо указать настройки Modbus-сервера (Рис. 1 на стр.57) и его области памяти (Рис. 2 на стр.57).

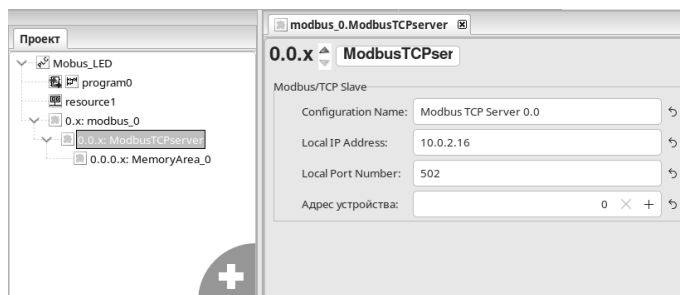


Рис. 1: Настройка Modbus-сервера

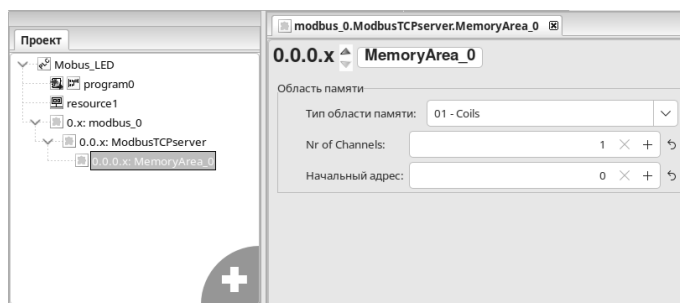


Рис. 2: Область памяти для Modbus-сервера

Реализация программы представлена на рис. 3 на стр. 58.

После нажатия на кнопку Build в Beremiz Получаем в указанной директории три файла:

- MB_0.h;
- plc.st;

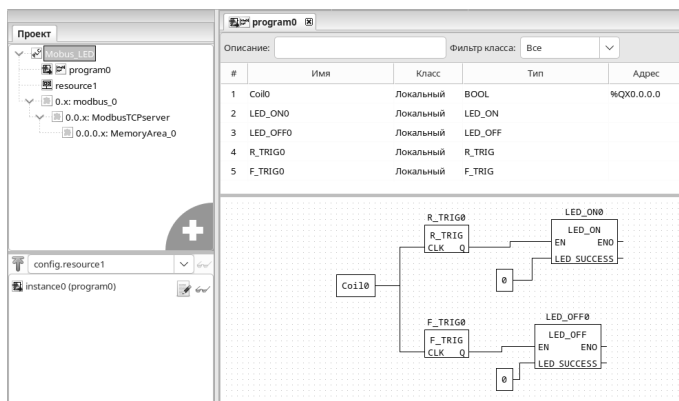


Рис. 3: Программа в VeriZim

- Mybuild.

Включаем их в сборку Embox. Данная сборка запускается на Qemu. И с локального терминала посылаются команды на включение и выключение светодиода по Modbus. Результат представлен на рис. 4 на стр. 58.

```
[info] (unit) initializing embox.fs.buffer_cache
[info] (unit) initializing embox.driver.flash.flash_nofs
[info] (unit) initializing embox.driver.tty.serial
[info] (unit) initializing embox.compat.posix.util.nanosleep
[info] (unit) initializing embox.driver.mmc.pl181
[info] (unit) initializing embox.driver.net.loopback
[info] (unit) initializing embox.driver.serial.pl011.tty50
[info] (test) running embox.lib.breakpoint.test.sw.breakpoint.test
[info] (test) running embox.libc.test.fpu_context_consistent
cy_test
[info] (test) running embox.test.posix.getopt_test
[info] (test) running embox.test.posix.enviro_test
[info] (test) running embox.test.stdlib.qsort_test
[info] (test) running embox.test.stdlib.bsearch_test
[info] (test) running embox.test.posix.popen_test
[info] (test) running embox.test.posix.pipe_test
[info] (test) running embox.test.posix.select_test
[info] (test) running embox.test.posix.poll_test
[info] (test) running embox.test.stdio.printf_test
[info] (test) running embox.test.recursion
[info] (test) running embox.test.critical
[info] (unit) initializing embox.fs.rootfs.dvfs
[info] (mod) runlevel is 2
[info] (mod) runlevel is 3
Default ID device/tty00
>export PWD=/
>export HOME=/
>netmanager
>tlsh
root@embox:/#pic_run
[info] (ledrv_stub) LED0 := 0
[info] (ledrv_stub) LED0 := 1
[info] (ledrv_stub) LED0 := 0
[info] (ledrv_stub) LED0 := 1
[info] (ledrv_stub) LED0 := 0
```

Рис. 4: Результат работы

Результат

В работе показана возможность использования СПО для построения систем АСУ-ТП в частности возможности реализовывать специализированное прикладное программное обеспечение на языках МЭК с помощью *Beremiz* и запускать на открытой ОС *Embox*

Дальнейшая работа была посвящена расширению возможностей операционной системы ОС РВ *Embox*⁴ для применения в сфере автоматизированных систем управления технологическими процессами (АСУ ТП). Благодаря своим особенностям, таким как модульность, минимальное потребление ресурсов и возможность тонкой настройки под конкретные задачи и при дальнейшем развитии этого направления, включая интеграцию с современными промышленными протоколами, *Embox* представляет собой перспективное решение для использования на предприятиях.

Литература

- [1] *Beremiz* Сайт проекта Beremiz. <https://beremiz.org/>
- [2] *Embox* Сайт Embox. <https://emboxing.ru/>
- [3] *Wikipedia* IEC 61131-3. https://en.wikipedia.org/wiki/IEC_61131-3
- [4] *Github* Репозиторий Matiec. <https://github.com/nucleron/matiec>

Леонид Чашкин, Сергей Полесский, Сергей Тумковский
Москва, НИУ ВШЭ
https://github.com/lbchashkin/digital_twin

Построение моделей цифровых двойников изделий электронной техники

Аннотация

Цифровой двойник изделия — система, состоящая из цифровой модели изделия и двусторонних информационных связей с изделием или его составными частями [1].

⁴Работа велась в отдельной ветке по ссылке <https://github.com/embox/embox/tree/softplc-modbus>.

Ключевые слова: *цифровой двойник, модель, свободное программное обеспечение.*

В настоящее время актуальным направлением развития остаётся создание цифровых двойников изделий с высокой точностью (более 95%) [2]. Актуальность цифровых двойников связана с возможностью их использования на всех этапах жизненного цикла изделия.

В первую очередь это позволяет уменьшить издержки на проводимые испытания изделия электронной техники. Более того, внедрение цифровых двойников позволяет обеспечить конкурентоспособность производимых изделий и повысить скорость их вывода на рынок.

Многие компании уже начинают создавать модели цифровых двойников изделий различной сложности.

Для увеличения скорости и точности создания моделей цифровых двойников требуется разработка системы для проведения виртуальных испытаний. Данная система в первую очередь должна позволять связывать цифровые модели и изделия электронной техники, а также накапливать, проводить анализ и сравнение получаемых данных.

Целью разработки данной системы является повышение достоверности имитационных моделей изделий электронной техники с учётом электрических, тепловых характеристик и надёжности.

Алгоритм

Алгоритм построения цифрового двойника включает следующие шаги:

1. Создание цифровой модели изделия
2. Создание связи изделия с цифровой моделью
3. Получение данных с изделия
4. Верификация модели на основе реальных данных

Острой проблемой здесь остаётся поиск и создание качественной цифровой модели изделия, которая позволяла бы с наибольшей точностью имитировать работу реального устройства.

Частично данная проблема решается за счёт проводимой верификации модели и корректировки её параметров, однако в связи с невозможностью измерения абсолютно всех характеристик каждого узла цепи данная верификация позволяет лишь ненамного повысить

точность будущего цифрового двойника, но не сможет заменить качественную цифровую модель.

После верификации имитационной модели происходит работа цифрового двойника, которая состоит из получения данных с изделия, расчёта модели, сравнения показателей, вычисления целевых функций, анализа результатов.

Архитектура системы

Архитектура системы для проведения виртуальных испытаний с использованием цифровых двойников представлена на рисунке 1. Модули системы написаны на языке программирования Python [5].

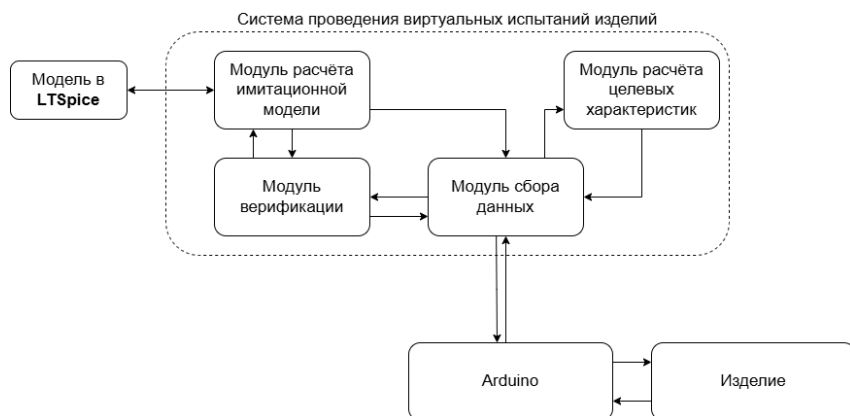


Рис. 1: Архитектура системы.

Связь датчиков изделия с системой происходит через платформу Arduino. Данные передаются через последовательный Serial порт.

Система состоит из 4 основных модулей:

1. Модуль сбора данных осуществляет взаимодействие с платформой Arduino, а также с другими модулями для накопления полученных данных и расчётных характеристик
2. Модуль верификации позволяет приблизить характеристики имитационной модели к характеристикам изделия

3. Модуль расчёта имитационной модели взаимодействует с моделью, построенной в программном обеспечении LTSpice, с помощью библиотеки PyLTSpice [3] языка программирования Python
4. Модуль расчёта целевых характеристик позволяет рассчитывать заданные целевые электрические характеристики изделия и его модели

Результаты

В настоящий момент проведено тестирование системы на примере транзисторного усилительного каскада по схеме с общим эмиттером (рисунок 2).

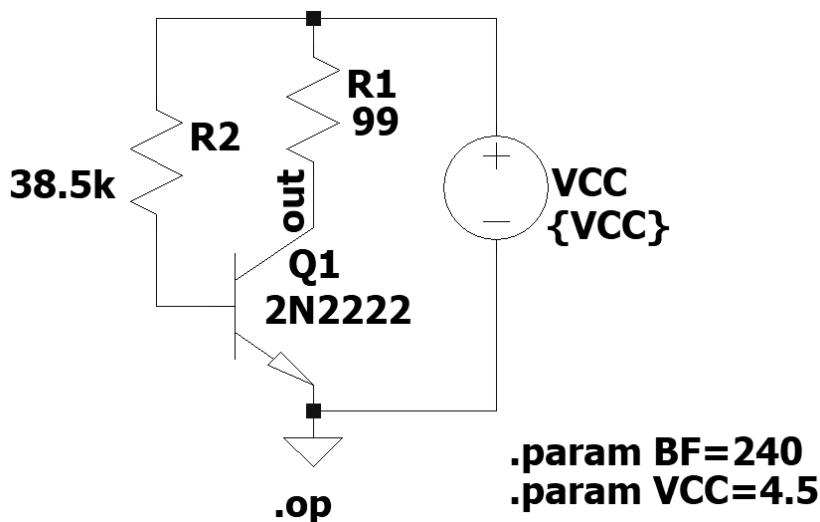


Рис. 2: Электрическая схема транзисторного усилительного каскада.

Результаты моделирования представлены на рисунке 3. Результаты были обработаны с помощью сглаживающей функции. Результаты с изделия (на макете) (верхняя кривая) отличаются от результатов модели (нижняя кривая) не более чем на 0,006 В. Данная разница может быть вызвана погрешностью датчиков, а также ограничениями платформы Arduino при измерении напряжения.

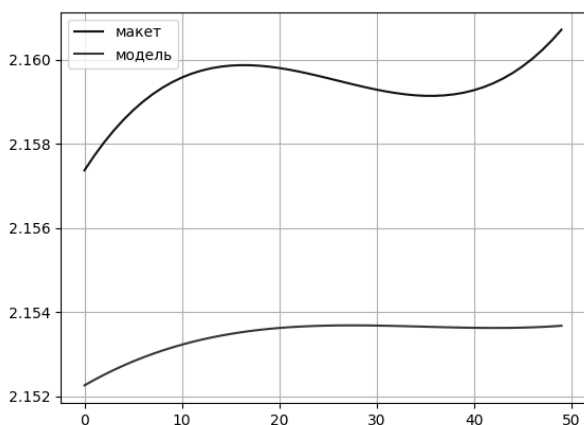


Рис. 3: Результаты моделирования.

Таким образом, на данной схеме погрешность модели составляет около 0,3%.

Однако столь низкая погрешность может быть вызвана верификацией модели, во время которой изменяются характеристики транзистора, чтобы минимизировать разницу с реальным изделием. Для более точной оценки достоверности модели цифрового двойника требуется учёт большего количества параметров, в том числе, например, тепловых характеристик. Также точность может понизиться при создании цифрового двойника комплексных изделий с большим количеством характеристик.

Перспективы развития

В дальнейшем планируется обработка более комплексных имитационных моделей. Для увеличения точности цифрового двойника возможно добавление измерения тепловых характеристик с проведением теплового моделирования.

В качестве целевых показателей необходимо добавление параметров надёжности, для расчёта которых может быть использовано программное обеспечение АСОНИКА [4].

Разрабатываемая система позволит агрегировать и собирать данные о виртуальных испытаниях различных изделий, а также их имитационных моделей.

Накапливаемые данные могут в дальнейшем быть использованы для создания модели нейронной сети для динамического анализа характеристик изделия и имитационной модели с целью автоматического выявления проблем на компонентах изделия.

Литература

- [1] ГОСТ Р 57700.37–2001 «Компьютерные модели и моделирование. Цифровые двойники изделий. Общие положения»
- [2] Семёнова К. В. *Разработка цифровых двойников силовых трансформаторов* / К. В. Семёнова, А. И. Тихонов, И. С. Снитко, А. В. Подобный, А. А. Каржевин // Надёжность и долговечность машин и механизмов. — Иваново, 2020. — С. 307–311.
- [3] Библиотека PyLTSpice, <https://github.com/nunobrum/PyLTSpice> (дата обращения 16.01.2025)
- [4] Программный комплекс АСОНИКА-К, <https://asonika-k.ru/company> (дата обращения 16.01.2025)
- [5] Исходный код проекта, https://github.com/lbchashkin/digital_twin (дата обращения 21.01.2025)

Александр Лакиза, Виталий Бондаренко, Диана Лакиза
Донецк, Институт Проблем Искусственного Интеллекта, Донецкий
Государственный Университет

Разработка стенда для обучения студентов системам обнаружения вторжений

Аннотация

В докладе рассматривается процесс создания стенда для моделирования сети с использованием системы симуляции GNS3 для обучения системам обнаружения вторжений. Кратко рассмотрены системы моделирования сети. В качестве инструментов выбраны GNS3, Mikrotik CHR, Debian 12, Snort.

Ключевые слова: *Debian, GNS3, Snort*

Интернет с момента своего создания в 1969 году в виде ARPANET значительно изменился. Изначально он использовался лишь для связи институтов и военными, но на сегодняшний день доступ к сети Интернет занимает ключевое место в жизни человека и деятельности предприятий. Затронуты практически все сферы деятельности человека: торговля, медицина, банковское дело, связь с государственными органами, обучение.

Однако, вскоре после появления интернета в массах, злоумышленники стали искать способы использования цифровизации в своих целях. Будь то спортивный интерес, жажда наживы или нанесение ущерба конкуренту. На данный момент взломы также приобрели политический характер, в частности, с целью шпионажа, манифестаций, нанесения ущерба (а именно, уничтожение данных).

Эти примеры подчёркивают, что, несмотря на давнюю историю развития информационной безопасности, присутствует острая необходимость в квалифицированных кадрах. Специалист получает квалификацию по большей части посредством практического применения знаний. В реальной сети, которая находится в эксплуатации, проводить эксперименты и обучение является дурным тоном, а в некоторых случаях и просто запрещено из соображений безопасности. Поэтому важным этапом в обучении или исследовании новых технологий является создание различных моделей сети в целом или отдельных её участков.

На данный момент создано достаточно много систем симуляции сети [1]. На данный момент популярными являются:

- Cisco packet tracer;
- EVE-NG;
- GNS3.

Каждая из названных систем симуляции сети имеет свои преимущества и недостатки. Так, Cisco packet tracer и EVE-NG проприетарные, Cisco packet tracer поддерживает только оборудование Cisco, но у обеих хорошая поддержка. GNS3, в свою очередь, — полностью свободный продукт. Однако, он имеет свои технические недостатки и слабую поддержку, что характерно для некоммерческих продуктов.

Так как GNS3 относится к свободному ПО и обладает обширными возможностями по виртуализации, было принято решение использовать его в создании стенда. Дополнительным преимуществом является возможность собрать простую сеть на любом современном компьютере и проводить локальные эксперименты, что освобождает вычислительные ресурсы на серверах учебного заведения.

Для стенда использовался персональный компьютер с операционной системой debian 12 и графической оболочкой KDE. Установка проводилась согласно официальной инструкции [2]. Из исходных кодов были собраны ubridge и vpcs.

Ubridge — приложение для создания мостов между различными средами передачи данных [3]. Например, ethernet-wifi, ethernet-UDP туннель. В GNS3 используется в том числе для организации связи между виртуальными устройствами.

VPCS — симулятор виртуального персонального компьютера [4]. Представляет из себя приложение в котором реализован базовый функционал компьютеров. Например, возможность назначить IP адрес, запустить пинг, трассировку, проверить DNS.

После установки системы симуляции была создана модель сети. Сеть состоит из маршрутизатора, двух коммутаторов, четырёх виртуальных ПК, одной IDS и одного web сервера. Модель сети представлена на рисунке 1.

В качестве маршрутизатора был выбран Mikrotik Cloud Hosted Router — образ операционной системы RouterOS для серверов виртуализации. В качестве ОС для IDS и WEB-сервера — debian 12.

Рассмотрим настройку Mikrotik. На нём создано 4 физических интерфейса. Один — для доступа в интернет, второй для первой локальной сети, третий — для второй, а четвёртый для web сервера. Были

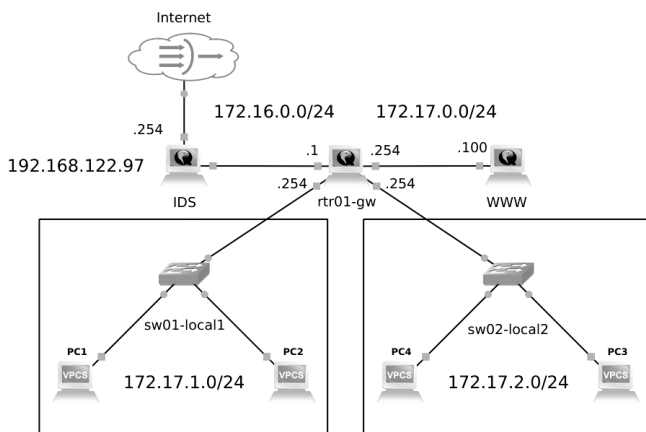


Рис. 1: Схема участка сети

назначены адреса на физические интерфейсы, созданы пулы адресов для dhcp, настроена работа DNS и NTP, создан сервер dhcp.

Настроены экземпляры VPCS на получение адреса по dhcp с помощью команды: `ip dhcp`

Существенным этапом создания стенда является настройка систем на базе debian 12. Инсталляция необходимых пакетов для работы на debian 12 проведена с использованием утилит создания сценариев для автоматизации задач ansible-playbook. В качестве web-сервера установлен nginx из репозитория.

В качестве свободной IDS установлен Snort [5] путём сборки из исходных кодов с адаптациями под debian 12.

Так как данная IDS является NIDS, то этот сервер будет являться также маршрутизатором. Была включена маршрутизация в линуксе, добавив `net.ipv4.ip_forward=1` в конец файла `/etc/sysctl.conf`. После чего для проверки работы системы использовали правило:

```
alert icmp any any -> $HOME_NET any (msg:"ICMP detected!"; sid:10000001; rev:001;)
```

Для проверки корректности конфигурации snort используем:

```
sudo snort -T -c /etc/snort/snort.conf
```

Если конфигурация корректна, то можем запускать систему:

```
sudo snort -A console -i ens4 -u snort -g snort -c /etc/snort/snort.conf
```

Запускаем пинг извне сети на защищаемый хост:

```
ping 172.16.0.1
```

В терминале стали появляться записи вида:

```
01/10-12:42:08.317350 [**] [1:10000001:1] ICMP detected! [**] [Priority: 0] {ICMP} 192.168.122.1 -> 172.16.0.1
```

Таким образом, система функционирует и захватывает пакеты. Следовательно, создание корректно функционирующего стенда можно считать завершённым.

Литература

- [1] Merion, Топ 5 инструментов моделирования сетей в 2020 году, URL: <https://wiki.merionet.ru/articles/top-5-instrumentov-modelirovaniya-setej-v-2020-godu>
- [2] GNS3, GNS3 Linux Install, URL: <https://docs.gns3.com/docs/getting-started/installation/linux/>
- [3] GNS3, uBridge, URL: <https://github.com/GNS3/ubridge>
- [4] GNS3, VPCS, URL: <https://github.com/GNS3/vpcs>
- [5] Cisco, Snort, URL: <https://www.snort.org/downloads>

Игорь Воронин, Ростислав Воронин, Евгений Коцюба
Шатура, МО, ИПЛИТ — филиал НИЦ Курчатовский институт

Проект: SmartTherm <http://smarththerm.ru/>

Разработка алгоритмов программного обеспечения для управления отоплением на основе ALT Linux

Аннотация

Контроллер Smart Therm на основе протокола Open Therm позволяет организовать эффективное управление и оптимизацию затрат для стабильной поддержке теплового режима внутри автономно отапливаемых помещений. Данное решение не требует дополнительных постоянных затрат и основано на свободном программном обеспечении от ALT Linux.

Ключевые слова: *интернет вещей, контроллер управления, свободное ПО, исследование, автоматизация.*

В жилых и производственных помещениях с автономным отоплением остро встаёт вопрос экономии энергоресурсов для отопления при стабильности поддержания наиболее оптимальных, комфортных температур. Для решения этой технологической задачи оптимизации управления тепловыми режимами отопления, позволяющих осуществлять ресурсо-энергосбережение самым оптимальным будет использование контроллера SmartTherm (ST) который общается с котлом по протоколу OpenTherm (OT) [1, 2]. Конечно это возможно, при условии наличия интерфейса связи OT на плате управления самого котла отопления.

Разработанные алгоритмы управления позволяют повысить тепловую комфортность помещений и энерго-ресурсосбережение комплексов отопления на 12–17% от годовых затрат на топливо. Результаты экспериментальных исследований водяного газового теплового комплекса для достижения необходимых показателей приведены в работе [3].

Для реализации энергоэффективного управления отоплением был разработан аппаратно программный комплекс SmartTherm(ST) с погодозависимой логикой. Он в себя включает собственно сам контроллер управления для связи с котлом по шине OpenTherm. Центральным процессором этого контроллера был выбран ESP WROOM-32. Кроме поддержки OpenTherm контроллер может измерять температуру с двух цифровых выносных датчиков температуры DS18B20 и обрабатывать один аналоговый сигнал. ST сам контролирует техническое состояние котла и сигнализирует о его неисправности, при возникновении внештатных ситуаций или отключении напряжения питания.

ST использует открытое программное обеспечение для прошивки контроллера и открытую электрическую схему [1]. Поскольку контроллер SmartTherm использует процессор ESP32 то возможна дистанционная связь с ним по встроенным модулям WiFi/Bt. Самую последнюю бинарную версию прошивки можно скачать с гитхаба [6]. Либо же из исходников её собрать самостоятельно в среде Arduino и/или Platformio.

Обновление и загрузка прошивки контроллера в сам модуль ESP32 возможно как по кабелю USB, так и через WiFi из браузера.

Для загрузки бинарного файла в отладочную плату ESP WROOM-32 используется утилита ESPTool, которая устанавливается в ALT Linux командой:

```
# apt-get install esptool
```

После скачивания прошивки загрузить её в модуль ESP32 можно командой:

```
$ esptool write_flash 0x00000 Smart_Therm_0.1.2_20250118.bin
```

Оригинальная прошивка SmartTherm [6] позволяет подключить контроллер к отопительному котлу и диагностировать возможные проблемы как с котлом, так и со связью по WiFi. Благодаря разработанным алгоритмам, прошивка ST реализует систему управления температурным режимом с погодозависимой логикой, на основе пропорционально-интегрально-дифференциального (ПИД) регулятора. Данные о внешней среде можно получать от Яндекс-погоды или бесплатно из приложения HomeAssistant [5] от интеграции Forecast Home Норвежского метеорологического института.

В текущем релизе реализована система удалённого управления котлом через собственное облако, которое собирается и устанавливается на любом Linux хостинге [7] командой:

```
$ g++ Server2.cpp SmartServer2.cpp SmartServer.cpp  
    SmartClient.cpp AutoConfig.cpp SmartDevice.cpp  
    TCPconnection.cpp -pthread -o Smartserver
```

Запускается облачный сервер командой:

```
$ Smartserver &
```

В приложении для OS Android предусмотрена настройка на облачный хостинг по умолчанию. Преимущество этого облачного приложения в том, что оно не привязано ни к какому поставщику аппаратных решений, ни к какому хостинг — провайдеру и может быть запущена на любом linux сервере — без жёстких привязок к поставщикам программного обеспечения. Необходимым условием, для доступа к ресурсу извне является, только лишь наличие белого (реального) IP адреса на нём.

Приложение для смартфона на базе OS Android опубликовано на гитхабе под лицензией MIT license [8].

Приложение Home Assistant [5] было выбрано постольку, поскольку оно позволяет оформить удобное и наглядное управление с графиками и просмотр истории событий. Связь ST с НА и передача данных происходит по протоколу MQTT. Так же в НА разработаны сценарии автоматизации, что очень удобно для выполнения рутинных операций. Установка и настройка MQTT брокера, осуществляется в ALT Linux командой:

```
# apt-get install mosquitto
```

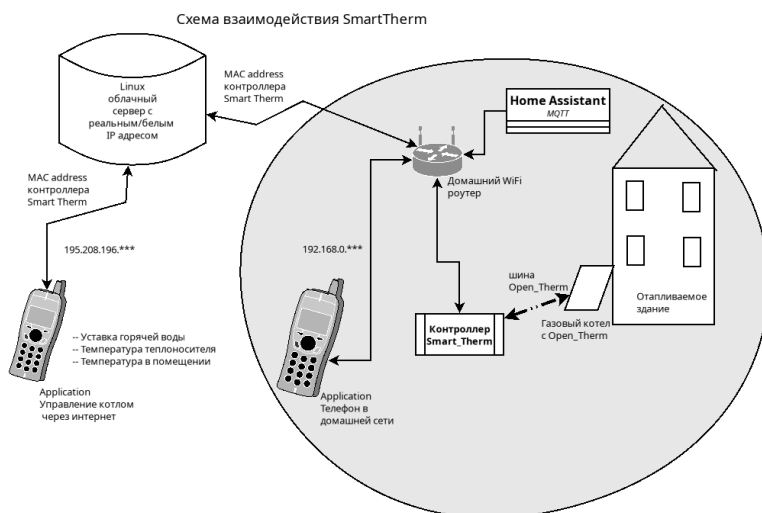


Рис. 1: Схема взаимодействия контроллера SmartTherm с внешними и локальными приложениями.

Литература

- [1] Аппаратная схема шлюза OpenTherm состоит из пяти основных частей: <https://otgw.tclcode.com/schematic.html>
- [2] Программная часть протокола OpenTherm https://ihormelnyk.com/Content/Pages/opentherm_library

- [3] Кашинский А. Н. *Разработка структурно-алгоритмического обеспечения и повышение эффективности управления процессом стабилизации температуры воздуха в автономно отапливаемом производственном помещении* / Автореферат диссертации на соискание учёной степени кандидата технических наук. Специальность 05.13.06 — «Автоматизация и управление технологическими процессами и производствами (промышленность)» // <https://sci.vlsu.ru/main/zaschita/Kashinskiy/Kashinskiy.pdf>
- [4] Описание контроллера SmartTherm <http://smarththerm.ru/>
- [5] Ресурсы HomeAssistant <https://www.home-assistant.io/>
- [6] Прошивка для контроллера SmartTherm <https://github.com/Evgen2/SmartTherm>
- [7] Облачный сервер <https://github.com/Evgen2/SmartServer>
- [8] Клиент для Android <https://github.com/Evgen2/SmartThermClient>

Леонид Меркин

Санкт-Петербург, НИУ ВШЭ в Санкт-Петербурге, Департамент информатики

Проект: SpaceBallistics <https://github.com/LMerkin/SpaceBallistics>,
<https://github.com/LMerkin/DimTypes>

Современный C++ для высоконадёжных вычислений в задачах космической баллистики

Ключевые слова: *C++23, шаблонное мета-программирование, размерные типы, численные методы, космическая баллистика, междисциплинарные образовательные программы, уровни зрелости ПО.*

Процесс разработки программного обеспечения (ПО) можно охарактеризовать различными *уровнями зрелости*. Например, широко известна классификация уровней зрелости, в основе которой лежит управление стоимостью одной строки программного кода [1]. Для наших целей можно предложить более простую, хотя и достаточно условную классификацию, в которой уровни зрелости соответствуют категориям от «Coding» (уровень 1) до «Computer Science» (уровень 5). Характеристики этих уровней приведены в Таблице 1.

Уровни 1 и 2 являются по сути до-индустриальными. Большинство современных индустриальных IT-решений соответствуют уровню 3.

№	Уровень	Типичный образовательный уровень разработчиков	Методы управления качеством ПО	Степень повторного использования ПО	Типичные применяемые технологии	Типичные области применения
1	Coding	Нет/среднее специальное образование	Практически отсутствуют	Отсутствует	PHP	Малые Web-сервисы
2	Programming	Среднее специальное образование или бакалавриат	Ad hoc ручное тестирование	Минимальная (в основном Copy + Paste)	Java Script, Python	Средне-масштабные Web-сервисы
3	Software Development	Бакалавриат по специальности	Автоматическое и ручное тестирование, Test-Driven Development, code reviews	Небольшая (менее 30%)	То же + Java, Rust	Электронная коммерция, крупномасштабные Web-сервисы
4	Software Engineering	Магистратура по специальности	Формальная спецификация требований, строгое типирование, Design by Contract, символьное выполнение, анализ временных свойств ПО, автоматическая генерация тестов	Целенаправленный дизайн для повторного использования (до 70 %)	Современный C++, Haskell, OCaml, Ada	Телекоммуникационные системы, финансовая индустрия, blockchain платформы
5	Computer Science	Учёная степень или магистратура по специальности	Формальная верификация функциональных и временных свойств ПО	Автоматическая генерация кода из формальных спецификаций	Spark, Ada, B, Isabelle/HOL, Z3	Ядерная, аэрокосмическая, оборонная промышленность; высокоскоростной транспорт

Таблица 1: Классификация уровней зрелости процесса разработки ПО

Однако для определённых областей применения, в первую очередь для критических по надёжности информационных инфраструктур и систем реального времени, согласно лучшим современным мировым практикам разработки ПО, требуются уровни зрелости 4 и 5. Уровень 4 («Software Engineering») соответствует требованиям надёжности ПО, нарушение которых может повлечь чрезвычайно большой материальный и иной ущерб; уровень 5 должен применяться тогда, когда нарушение свойств надёжности ПО может привести к человеческим жертвам или значительному экологическому ущербу.

К сожалению, в российской IT индустрии уровень 4 представлен явно недостаточно, а уровень 5 практически полностью отсутствует. Данные уровни мало применяются даже в тех областях, где они являются абсолютно необходимыми, например, в аэрокосмической про-

мышленности. Так, космический аппарат «Луна-25» потерпел аварию в августе 2023 года *«из-за возможного попадания в один массив данных команд с различными приоритетами»* [2], что означает непроведение, на этапе разработки бортовой системы управления, временного анализа задач и коммуникаций между ними. Такой анализ является стандартным для уровней зрелости 4–5.

Аналогичным образом, в российской системе высшего образования в сфере ИТ и прикладной математики, в настоящее время практически отсутствуют дисциплины, посвящённые задачам и методам разработки ПО с критическими уровнями надёжности.

С целью хотя бы частичного исправления данного положения, автор организовал преподавание методов разработки *высоконадёжного научного ПО* в рамках курса «Численные методы и методы оптимизации» по программе бакалавриата «Прикладной анализ данных» в Департаменте информатики НИУ ВШЭ в Санкт-Петербурге. Для этих целей были созданы проекты *DimTypes* и *SpaceBallistics*.

Наш подход соответствует уровню зрелости «4+»: мы не используем формально-логические методы верификации ПО, которые соответствовали бы уровню 5, так как этот уровень был бы чрезвычайно трудозатратным и практически недостижимым в рамках программы бакалавриата. Вместо этого, за основу были взяты современные стандарты языка C++ (включая C++23), которые позволяют, на основе парадигмы шаблонного мета-программирования, реализовать предметно-ориентированную систему типов для выбранной области научных вычислений — задач космической баллистики, т. е. расчётов траекторий космических аппаратов (КА) и ракет-носителей (РН).

Систематическое применение строгого предметно-ориентированного типирования позволяет детектировать *на этапе компиляции* достаточно широкий класс возможных семантических ошибок в рассматриваемом классе численных алгоритмов, без какого-либо ущерба для вычислительной эффективности.

В частности, проект *DimTypes* представляет собой *header-only* библиотеку, реализующую *размерные типы* в C++, т. е. вещественные (а также комплексные) типы, которые описывают физические величины, а потому дополнительно характеризуются физическими размерностями и системами физических единиц. Таким образом, система размерных типов предотвращает невалидные математические операции с размерными физическими величинами, например, сложение величин с различными размерностями или смешение систем физиче-

ских единиц (последняя проблема привела к аварии американского КА «Mars Climate Orbiter» в 1999 г. [3]).

В C++, как и в некоторых других языках программирования, уже существуют различные системы размерных типов, например `boost::units`. Однако `DimTypes` имеет ряд преимуществ перед существующими реализациями: наша библиотека позволяет пользователю при необходимости декларировать свои предметно-ориентированные размерности (всего до 9-ти); поддерживает рациональные (а не только целочисленные) степени в размерностях, что важно для задач космической баллистики; имеет эффективную реализацию, в которой вектора степеней размерностей (параметры C++ шаблонов) есть последовательности битовых полей в `uint64_t`, а битовые поля кодируют рациональные числа в арифметике Z_p . Например, для 9-ти размерностей используются 7-битовые поля, $p = 127$, что полностью достаточно для изоморфного кодирования всех встречающихся на практике степеней размерностей.

Библиотека `SpaceBallistics` реализована на основе `DimTypes`. Она, во-первых, реализует предметно-ориентированную систему типов для многочисленных систем координат и систем отсчёта времени, используемых в космической баллистике, например: системы, жёстко связанные с КА; топоцентрические системы, связанные с точкой старта; планетоцентрические системы с вращающимися осями (в силу осевого вращения планеты) и с неподвижными инерциальными осями; инерциальные барицентрические системы, связанные с центром масс Солнечной системы.

Во-вторых, все векторные величины в `SpaceBallistics` строго типированы: соответствующими физическими размерностями (через `DimTypes`), системами координат, а также перечислимыми типами объектов, к которым данные вектора относятся (через шаблонные параметры). Таким образом, уже на этапе компиляции исключаются ошибки, связанные с некорректными использованием векторов (например, смешение систем координат).

В-третьих, библиотека `SpaceBallistics` содержит набор физических моделей и математических методов, реализованных над вышеуказанной системой предметно-ориентированных типов, например: подмножество планетных эфемерид JPL DE440 [4], модели неоднородных гравитационных потенциалов Земли и Луны, модели осевого вращения Земли и Луны, прецессии и нутации земной оси, модель атмосферы Земли, строго-типированный интегратор для дифферен-

циальных уравнений поступательного и вращательного движения РН и КА, и др.

В-четвёртых, библиотека **SpaceBallistics** в максимально возможной степени использует вычисления на этапе компиляции, поддерживаемые современными стандартами языка C++. Таким образом достигается не только глубокая оптимизация генерируемого машинного кода, но также минимизация вероятности возникновения неопределённого поведения программы.

Библиотека **SpaceBallistics** находится ещё на этапе разработки, но уже сейчас обе библиотеки используются автором в преподавании *междисциплинарных* образовательных программ и реализации учебных проектов, объединяющих элементы прикладной математики (численные методы), инженерии высоконадёжного ПО (предметно-ориентированные системы типов в C++), астродинамики и физики.

По глубокому убеждению автора, принципиальное разделение между преподаванием физико-математических и IT-дисциплин, сформировавшееся в результате «Интернет-революции» 1990-х годов, далеко не всегда оправдано. Сфера разработки высоконадёжных систем управления кибер-физическими объектами, функционирующими в реальном масштабе времени, требует специалистов, имеющих как специальную (с высокими уровнями зрелости), так и интегративную подготовку. Организовать такую подготовку — важная задача современной российской высшей школы.

Литература

- [1] Chrissis, M. B., Konrad M., Shrum S. *CMMI for Development: Guidelines for Process Integration and Product Improvement*, 3ed. — Addison-Wesley, 2011
- [2] *Роскосмос назвал вероятную причину аварии «Луны-25»*. — URL: <https://tass.ru/kosmos/18896789>. Проверено 17.01.2025
- [3] *Mars Climate Orbiter Mishap Investigation Board Phase I Report*: November 10, 1999. — URL: https://llis.nasa.gov/llis_lib/pdf/1009464main1_0641-mr.pdf. Проверено 17.01.2025
- [4] Park, R. S. и др. *The JPL Planetary and Lunar Ephemerides DE440 and DE441* // The Astronomical Journal, стр. 161–105, март 2021

Николай Непейвода
Переславль-Залесский, ИПС РАН

Неполные и неточные задачи в обучении информатике

Аннотация

В реальности программист и информатик будет почти всегда (кроме работы в нескольких уникальных фирмах) получать в лучшем случае неполное, а часто ошибочное, «техническое задание». Худший случай, когда оно имеет при этом внешнюю форму полного и точного и оформлено по всем ГОСТам.

В практике обучения в УдГУ и на ФИТ НГУ применялись неточные и неверные постановки задач, чтобы отрабатывать навыки критического отношения к формулировкам и их уточнения формальными и неформальными методами. Это было применено также в олимпиадах (Всероссийской и Удмуртской). Поскольку это практиковалось достаточно длительное время, был накоплен некоторый опыт. Его систематическое изложение требует большой статьи, поэтому в тезисах ограничимся кратким анонсом.

Ключевые слова: обучение программированию, поиск ошибок, связь с реальностью.

При уточнении учебных задач поощрялись коллективные обсуждения студентов, но затем организовывалась соревновательная ситуация, когда разные варианты уточнения решали отдельные люди.

На олимпиадах задачи имели либо принципиально неформальную постановку (например, в 90-х годах обработка безграмотных отчётов об операциях работников шарашки, при условии невозможности дисциплинировать их на единую форму, а посаженная на обработку девочка просто исправляла орфографию и набивала текст в почти произвольной форме, сохраняя лишь ключевые поля, типа кто (не соблюдая официальных имён, например, Сергей Иванов мог быть и Серёгой, и Серым), кому, что, за сколько (тоже название валюты могло иногда быть сленговым) загнал либо купил. При этом соревнующийся заодно сам писал аналогичные тест для других, и выше всего оценивался тест, который сумела пройти примерно половина участников.

Также практиковались двухтактные задачи, когда в первый день участники получали почти точную формулировку, затем она прогонялась на тестах и каждый мог по результатам тестов её доработать.

А на следующий день участники получали на вид чуть изменённую и уточнённую задачу и *чужую* программу, которую должен был переработать, причём оценивалась заодно близость текста переработанной и исходной программы. Этот пункт заранее никогда не объявлялся.

Также практиковались «игровые» задачи, когда несколько модулей участников соревновались между собой под управлением общей программы. Здесь интерфейс модулей задавался всегда строго и полно. Зато правила игры часто были чуть неполны. И после анализа результатов первого тура давалась возможность доработать свои модули для второго тура, но неполноту правил должны были восполнить каждый из участников, проанализировав логи всех соревнований первого тура.

Главная трудность таких задач — большая сложность проверки и невозможность обеспечить формальную объективность. И очевидно, что проверка требует полной открытости и часто хорошей комментируемости кода программ участников.

Олег Игумнов, Андрей Михеев

Москва, РЭУ им. Плеханова, Финансовый университет, ООО «Процессные технологии»

Проект: HWChecker <https://github.com/RadiantDelta/HWChecker>

Разработка и использование в Финансовом университете свободной системы проверки контрольных и домашних работ на Java

Аннотация

Доклад продолжает тему, поднятую на 19 конференции разработчиков свободных программ в работе «Г. В. Курячий, В. А. Арефьев, Н. С. Барабанов. Организация рабочего процесса разработки системы проверки домашних заданий» [1]. В этой работе было предложено не использовать готовую систему проверки заданий типа ejudge, а силами студентов разработать небольшую лёгкую в настройке и использовании систему тестирования заданий, не требующую сложной системы безопасности, т. к. использовать её предполагается во внутренней среде ВУЗа, в которой все пользователи хорошо известны и заинтересованы в получении хороших оценок за свои работы. Однако, в представленном в работе проекте [2] производится проверка программ на Python, а

нам требовалась проверка программ на Java (Т.к. основной наш свободный проект RunaWFE написан на Java. Проект сотрудничает с ВУЗами, участники проекта ведут занятия в ВУЗах и привлекают в проект студентов). Попытка расширить на Java уже существующую систему тестирования на Python не оказалась успешной, т.к. её разработчики были загружены работой в других проектах. Поэтому из этой системы были взяты только идеи, а система тестирования на Java [3] была написана «с нуля», с привлечением студентов проекта RunaWFE. В докладе рассказано про разработку и использование системы в ВУЗе, о её достоинствах и недостатках, а также о борьбе с «Искусственным Интеллектом».

Ключевые слова: *СПО, Java, тестирование, домашние работы, контрольные работы.*

Постановка задачи

Количество студентов, у которых преподаватель ведёт семинары по программированию, как правило, достаточно большое (обычно, — от 50 до 200). Обязательным элементом занятий является самостоятельное выполнение заданий студентами, результатом которых являются разработанные программы. При таком количестве студентов преподаватель физически не может «вручную» полноценно проверить каждое выполненное задание. Поэтому используются различные упрощённые формы проверки работ, такие как: выборочная проверка, поверхностный просмотр кода без компиляции и запуска программы, коллективное решение и защита работ и т.п. Одной из форм проверки является автоматическое тестирование программ. Такой вариант проверки позволяет одному преподавателю работать с большим количеством студентов. У этого подхода тоже есть минусы. В частности, тесты обычно никак не оценивают качество кода программы. Но основной проблемой последних полутора лет преподавания становится использование студентами для генерации кода сервисов Искусственного Интеллекта (ИИ). Стандартные задачи ИИ решает уже достаточно хорошо и это приводит к тому, что оценки у студентов, пытающихся самостоятельно решить задачу, оказываются хуже, чем у студентов, представляющих решения, сгенерированные ИИ. Что заметно демотивирует студентов. Однако, оказалось, что пока ещё в формулировках задач можно «обмануть» ИИ так, чтобы он проявил элементы

своего «нечеловеческого мышления» и при помощи этого отделить самостоятельно разработанные программы от сгенерированных ИИ.

В Финансовом университете для загрузки результатов выполнения домашних и контрольных работ используется система на основе Moodle. В системе можно настроить ограничения по времени как для показа условий задач, так и для загрузки решений в систему для каждой группы (См. Рисунок 1).

Ограничение доступа

Ограничения доступа

☑Студент должен соответствовать любому из нижеследующих условий

Студент должен соответствовать всем из нижеследующих условий

Дата от 23 Сентябрь 2024 11 : 50

и

Дата до 23 Сентябрь 2024 13 : 20

и

Группа ID22-3

Рис. 1: Установка периода для загрузки решения задачи для группы.

В курсе по Java в качестве решения задачи студенты загружают в систему zip-архив, содержащий проект IntelliJ IDEA Community Edition, в котором решается поставленная задача.

После окончания срока загрузки работ преподаватель может посмотреть все решения (см. Рисунок 2), а также скачать все решения студентов в виде одного архива (см. Рисунки 3, 4).

Архив содержит набор папок, названия которых соответствуют ФИО студентов. Внутри каждой папки содержится zip-архив, содержащий проект с решением задачи.

Тестирующая система должна работать следующим образом: Для проверки каждого задания предусмотрен набор тестов, задаваемых файлами тестовых входящих и исходящих данных. Также должен

Контрольная работа 3

Открыто с: Понедельник, 2 Декабрь 2024, 11:50
Срок сдачи: Понедельник, 2 Декабрь 2024, 15:40

 КР3_ИД22-2.docx	2 Декабрь 2024, 13:59
 КР3_ИД22-3.docx	2 Декабрь 2024, 01:12

Изолированные группы

Все участники

Резюме оценивания

Скрыто от студентов	Нет
Участники	66
Ответы	30
Требуют оценки	30
Оставшееся время	Задание уже должно быть выполнено
Поступившие представления	Разрешено только для участников, которым было предоставлено продление срока.

Просмотр всех ответов

Оценка

Рис. 2: Просмотр всех ответов.

задаваться набор конструкций Java, которые студентам запрещено использовать (которые не входят в изучаемый курс). Тестирующая система должна собрать программу из исходного кода, для каждого студента и каждого теста поместить в соответствующее место тестовые входящие данные, выполнить программу и сравнить результат с тестовым исходящим файлом. В результате работы системы должен быть сгенерирован отчёт, в котором для каждого студента отмечено, какие тесты прошли, а какие нет и были ли использованы «запрещенные» конструкции.

Структура разработанной программы

Программа была реализована как студенческая работа в виде проекта, размещённого на github. Проект называется HW-checker, разработан на Java, имеет графический интерфейс на базе JavaFX и может быть собран с использованием Maven или запущен из IDE.

Контрольная работа 3

Открыто с: Понедельник, 2 Декабрь 2024, 11:50

Срок сдачи: Понедельник, 2 Декабрь 2024, 15:40

Действия оценивания

Скачать все ответы

Изолированные группы

ИД22-3

Имя Все А Б В Г Д Е Ж З И К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Э Ю Я

Фамилия Все А Б В Г Д Е Ж З И К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Э Ю Я

1

Изображение	Фамилия / Имя	Адрес электронной почты	Статус	Оценка	Редактировать	Последнее изменение (ответ)	Ответ в виде файла
Выбрать пользователя	Отчество	Логин	почты				
<input type="checkbox"/>	—	—	—	—	—	—	—

Рис. 3: Получение решений студентов.

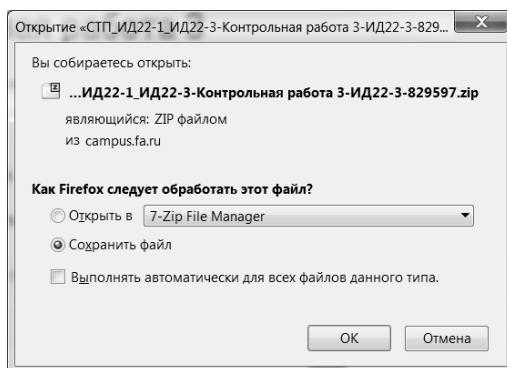


Рис. 4: Решения контрольной работы всех студентов группы в виде одного архива.

Программа анализирует проект каждого студента, компилирует и выполняет тестируемую программу на наборе тестов, используя класс CommandExecutor. Класс DocGenerator генерирует итоговый отчёт.

Основные компоненты программы:

- `HWCheckerApplication`: Главный класс приложения, отвечает за запуск программы.
- `Launcher`: Инициализация и подготовка программы к работе, осуществление проверки задания.
- `MainController`: Контроллер для управления основным интерфейсом.
- `SettingsController`: Контроллер для изменения настроек проверки.
- `Localization`: Поддержка локализации интерфейса.
- `DirDeleter`: Удаление временных файлов после проверки.
- `DocGenerator`: Генерация документации, связанной с проверкой.
- `Unzipper`: Распаковка архивов.
- `XmlEditor`: Работа с конфигурационными XML-файлами.
- `CommandExecutor`: Выполнение команд.
- `LaunchInfo`: Параметры проверки.
- `Report`: Отчёт о проверке.
- `TestInfo`: Управление информацией о тестах.
- `RestrictionChecker`: Анализирует проект на предмет соответствия заранее заданным правилам (например, использование запрещённых методов).

Конфигурация системы содержит следующие настройки:

- Использование Maven: Включить или отключить сборку проекта с использованием Maven.
- Использование H2-драйвера. — Определяет, требуется ли подключение находящегося в системе драйвера для H2-базы данных при выполнении тестов.
- Выбор версии JDK. — Система поддерживает выбор версии JDK для проверки работы. Это достигается благодаря возможности указания пути к конкретному JDK.
- Установка времени ожидания выполнения программы. — Для предотвращения зависания проверки из-за бесконечных циклов или длительного выполнения программы, система позволяет настроить максимальное время выполнения теста.

Работа тестирующей системы

Для работы тестирующей системы в файловой системе должны быть созданы три папки: Restrictions, Students и Tests (см. Рисунок 5).




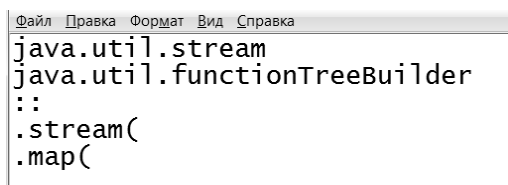
Имя	Дата изменения	Тип
 Restrictions	06.10.2024 15:53	Папка с файлами
 Students	06.10.2024 16:13	Папка с файлами
 Tests	06.10.2024 16:07	Папка с файлами

Рис. 5: Папки, необходимые для работы программы.

В папку Restrictions надо поместить файл, содержащий список запрещённых конструкций. Пример файла приведён на Рисунке 6.



```
Файл  Правка  Формат  Вид  Справка
java.util.stream
java.util.functionTreeBuilder
::
.stream(
.map(
```

Рис. 6: Пример файла с запрещёнными конструкциями (запрещается использование стримов).

В папку Students надо разархивировать файл-архив с решениями студентов.

В папку Tests надо поместить набор папок, названия которых совпадают с названиями заданий (например, HW1, HW2 и т.д.). Внутри каждой папки должны содержаться папки, соответствующие тестам (например, TEST1, TEST2 и т.д.). Внутри папок тестов находятся папки INPUT и OUTPUT. Для каждого теста задаются файл (файлы) входящих данных и файл исходящих данных. Например, input.csv и output.txt. Файлы входящих данных находятся в папке INPUT, а исходящих — в папке OUTPUT.

На текущий момент в системе реализованы тесты двух типов. В случае тестов первого типа система компилирует программу из исход-

ного кода при помощи `jdk`, в случае тестов второго типа используется `maven` и генерируется `jar`-файл, который запускается на выполнение.

После запуска системы тестирования в появившемся графическом окне надо настроить путь к `jdk` командой `Settings`, настроить пути к папкам, выбрать тип используемых тестов и указать, надо ли использовать содержащийся в системе драйвер к СУБД `H2`. (см. Рисунок 7).

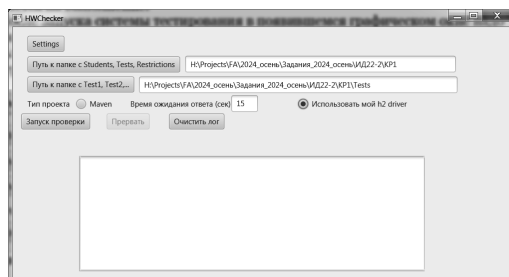


Рис. 7: Графическое окно системы тестирования.

Далее надо выполнить команду «Запуск проверки». После окончания тестирования будет сгенерирован отчёт о тестировании.

Применение тестирующей системы. Борьба с «Искусственным Интеллектом»

Использование разработанной системы позволяет быстро проверять большое количество работ студентов. Для определения заимствованных решений и решений, сгенерированных сервисами ИИ, применялось следующее:

- В качестве задач в основном давались классические задачи на разработку алгоритмов и структур данных, но к ним делались дополнения, не являющиеся логичными с точки зрения «здорового смысла» и практического использования.
- Также была задействована проблема, для которой не существует решения, однако существует большое количество заметно отличающихся решений для её частных случаев.
- Использовались достаточно длинные формулировки заданий.

- Перед тем, как давать задачу студентам, проверялось, что ChatGPT решает её неправильно.
- Правильность загруженных в систему решений не сообщалась студентам до окончания времени загрузки.
- Искались решения с одинаковыми, нетипичными с точки зрения обычной логики ошибками.

Пример определения заимствованных решений представлен на Рисунке 8.

```

- 7,9,4,1,2,8,5,6,3 - аннулировано
- 7,4,9,2,1,8,6,5,3 - аннулировано
- 7,9,4,1,[2,8],5,6,3 Допуск к защите
- 7,4,9,2,1,8,6,5,3 - аннулировано
- 7,4,9,2,1,8,6,5,3 - аннулировано
- 7,4,9,2,1,8,6,5,3 - аннулировано
- 7,9,6,4,2,1,3,8,5 Допуск к защите
- 7,[,9,4,2,1,6,8,5,3] Допуск к защите
- 7,9,4,1,2,8,5,6,3 - аннулировано
- 7,9,4,1,2,8,5,6,3 - аннулировано
- 7,9,1,4,2,6,8,5 Допуск к защите
- 7,4,9,2,1,8,6,5,3 - аннулировано
- 7,9,4,1,2,[6,8],5,3 - аннулировано
- 7,9,4,1,2,[6,8],5,3 - аннулировано
- 7,4,9,2,1,8,6,5,3 - аннулировано
- 7,9,4,1,2,8,6,5,3,10 Допуск к защите
- 7,4,9,2,1,8,6,5,3 - аннулировано

```

Рис. 8: Определение заимствованных решений. (Правильное решение: 7,9,4,1,2,8,6,5,3).

Выводы

В работе [4] проверялись правильные решения на «похожесть». В данной работе все правильные решения (т.е. прошедшие тесты) принимались, заимствованные — аннулировались, а для неправильных, но не заимствованных, давалась возможность исправить и защитить их на следующем семинаре.

В целом, можно согласиться с выводом работы [4]: Пока ещё, применяя различные приёмы, можно более-менее «отлавливать» заимствованные решения, но в будущем это будет всё более сложно и менее точно и в конечном счёте ИИ и другие технологии разовьются

настолько, что сделать это (в той форме обучения, которая используется сейчас) будет нельзя. Кроме того, в обучении программированию появился «запрос от бизнеса»: учить студентов программированию «вместе с ИИ» в таких средах, как Cursor и GitHub Copilot — это приводит к большей продуктивности программиста и, соответственно, к существенному удешевлению разработки. По-видимому, относиться к такой ситуации надо так же, как к обучению на уроках математики в школе: в младших классах пользоваться калькулятором нельзя и считать надо самому, а в средних и старших, калькулятор — это необходимость и при решении задач надо уметь им хорошо пользоваться.

При этом появляется ещё одна проблема, которой не было в прошлом. ВУЗовское образование должно готовить обучающихся к решению востребованных задач, т.е. таких задач, которые не были решены раньше. Основной приём, применяющийся при подготовке студентов к этому, состоит в том, что преподаватели дают студентам постепенно усложняющиеся наборы задач, которые преподаватели знают как решать, а студенты — нет. При этом студенты заново решают эти задачи, повторяя действия тех, кто решил их когда-то в первый раз. В настоящее время ИИ умеет быстро искать уже существующие решения, но не умеет решать новые задачи. То есть, студенты, сдававшие задачи только с помощью ИИ, не смогут решать новые задачи.

Лучшее решение, конечно, — это сознательность студентов: чтобы они не искали «косвенных способов» решения учебных задач, а относились к ним как к ещё не решённым задачам. Однако, в существующих условиях таких студентов не будет много.

Литература

- [1] Курячий Г. В., Арефьев В. А., Барабанов Н. С. *Организация рабочего процесса разработки системы проверки домашних заданий*. — в кн.: Десятнадцатая конференция разработчиков свободных программ: Тезисы докладов / Переславль, 29 сентября — 1 октября 2023 г. — М.: МАКС Пресс, 2023. С. 14–17
- [2] Проект Hworker: <https://github.com/FrBrGeorge/HWorker>
- [3] Проект HWChecker: <https://github.com/RadiantDelta/HWChecker>
- [4] Курячий Г. В. *Как я делал проверку копипасты для спецкурса по Python3 и что из этого вышло*. // Тринадцатая конференция «Свободное программное обеспечение в высшей школе». Материалы конференции // Переславль, 26–28 января 2018 г. М.: Макс Пресс, 2018. С. 49–60

Никита Шалаев

Санкт-Петербург, ФМОПИ СЗИУ РАНХиГС

«Чему учить студентов», или анализ использования СПО в исследованиях

Аннотация

Внедрение СПО в обучение студентов в области социальных наук часто наталкивается на непонимание смысла такой миграции — многие люди привыкли пользоваться проприетарными программами и считают их неоспоримым эталоном профессионального инструментария, и аргументы об идеологических и лицензионных преимуществах СПО не выглядят убедительными. Однако есть и объективные аргументы в пользу СПО, прежде всего — его распространение среди практикующих эмпирических исследователей.

Ключевые слова: *Социальные науки, данные для репликации, эмпирические исследования.*

Среди участвующих в обучении студентов по специальностям, связанным с социальными науками, бытует расхожее мнение о существовании некоторого «индустриального стандарта» для программного обеспечения, использующегося в эмпирических исследованиях, и, следовательно, которому и нужно обучать студентов. При этом это ПО является коммерческим и проприетарным, но это не мешает рекомендовать его использование, в т.ч. с применением различных не вполне корректных схем, включая нарушение авторских прав. Ярким примером таких инструментов являются программы SPSS, SAS и STATA. Эти инструменты, в особенности SPSS, получили широкое распространение ещё в 90х годах, и использование даже их нелицензионных копий успело укорениться в сознании в качестве нормы.

Предложения о переходе к использованию СПО частенько встречаются со скепсисом. Перспективы оказаться без доступа к привычному проприетарному ПО не выглядят убедительно, а необходимость переучиваться никого не привлекает. Однако есть ли доводы в пользу обучения студентов инструментарию, относящемуся к СПО, не основывающиеся на соображениях лицензионного характера? Рейтинги популярности языков программирования или анализ предложений на

рынке труда едва ли могут раскрыть инструментарий исследователя-эмпирика. Например, с точки зрения рейтинга ТЮВЕ лидирует Python, а язык статистического программирования R находится на 18 месте, в рейтинге IEEE Spectrum (2024) и по популярности на StackOverflow (2023) R уже на 20 месте — и только в PYPL на 6. Ситуация выглядит так, что изучение R является делом довольно бессмысленным. Что же касается прикладных программ, то для них аналогичных рейтингов не сыскать вовсе.

ПО	Всего	FS	ZN	DR	MD	HD
Excel	23673	7276	4875	3534	1630	620
R	22264	8295	4005	4964	705	933
Python	10660	2164	4852	726	428	93
Matlab	7204	1883	1893	548	507	173
Stata	1730	69	62	12	69	1168
SPSS	1710	283	64	34	137	688
Java	1118	199	632	32	115	4
Perl	1027	401	138	254	26	7
Fortran	819	55	170	26	179	23
SAS	441	72	38	22	30	113
gnuplot	122	34	45	3	15	0

Таблица 1: Упоминания различного ПО в репозиториях

К счастью, эмпирические исследования в наше время оставляют после себя не только тексты публикаций, но и сопроводительные материалы, в том числе — данные для репликации. Разумеется, далеко не все исследования снабжаются данными для репликации в обязательном порядке, но такое действие к нашему времени стало настолько популярным, что существует целый набор репозиторий таких материалов. Кроме того, логично предположить, что хотя выборка получается смещённой, смещается она как раз в пользу наиболее технически продвинутых и уверенных в себе авторов, которые смело готовы открыть свои наработки для проверки всем желающим.

Анализ данных для репликации, размещённых в ведущих репозиториях, показывает, что именно СПО (например, язык R) в реальной

практике во многих сферах уверенно занимает лидирующие позиции, в частности, превосходя по популярности указанные проприетарные инструменты. Для исследования популярности ПО была собрана статистика по результатам поиска в соответствующих репозиториях по ключевым словам, покрывающим обычный спектр типовых выражений, использующихся при описании наборов данных.

В таблице 1 приведена избранная статистика по 5 репозиториям (FS = Figshare ARS, ZN = Zenodo, DR = DRYAD, MD = Mendeley Data, HD = Harvard Dataverse), в которых общее число результатов оказалось наибольшим, а также общее число результатов по всем репозиториям, данные с которых были агрегированы сервисом Mendeley. В качестве «эталонного варианта» выступает Excel — синоним как самого простого варианта для предоставления данных, и одновременно программного обеспечения, которое доступно практически повсеместно и выполняет большое число базовых аналитических операций.

Мы видим, что в отличие от показателей рейтингов популярности языков программирования, в целом R оказался вдвое популярнее Python, хотя из рассмотренных инструментов лидером, конечно, оказался Excel, олицетворяющий электронные таблицы в целом. А вот специализированные статистические программы (SPSS, SAS, STATA), якобы являющиеся «профессиональным стандартом», существенно уступают в популярности и R, и Python за исключением разве что Harvard Dataverse, где STATA является лидирующим инструментом. По-прежнему используются и такие специализированные языки, как Perl и Fortran. Примерно на уровне gnuplot и SAS находятся такие представители СПО, как JASP (статистика), pajek и gephi (сетевой анализ). Значительно выше популярность GNU Octave, а вот популярность системы символьной алгебры maxima, наоборот, мала.

В результате мы видим, что популярность «солидных» проприетарных программ для статистического анализа сильно переоценена, и на практике на первый план выходят представители СПО — как язык программирования общего назначения Python, так и специализированный язык статистического программирования R. Пользуются сравнимой популярностью и другие свободные программные продукты; и даже такой узко-специализированный инструмент, как gnuplot, находит своих благодарных пользователей. Это, в свою очередь, даёт в наши руки весьма весомый объективный аргумент в пользу отка-

за от проприетарного ПО в образовательном процессе. В наши дни именно СПО является «стандартным инструментарием» для исследователей. И именно работе с ним следует учить студентов, чтобы они приобщались к передовым тенденциям мировой науки.

Дмитрий Муканин, Роман Катунцев

Иркутск, ООО «Степплер»

Проект: Stappler SDK <https://stappler.dev>

Проблемы подготовки прикладных разработчиков в современной России

Аннотация

В современном мире прослеживаются чёткие тенденции: деградация технического образования, дефицит высококвалифицированных кадров и стремление компаний к снижению издержек. Именно по этому современные популярные технологии и фреймворки стараются снизить порог входа в разработку, что влечёт за собой тотальное снижение уровня компетентности программистов. Стремление к излишнему упрощению инструментов влечёт за собой и другие не менее важные проблемы. Чтобы наша страна была готова к решению этих проблем, нам необходимо создавать собственные открытые технологии и фреймворки, и, активно внедрять их в образовательный процесс для подготовки собственных высококвалифицированных инженеров-программистов.

Ключевые слова: *СПО, кроссплатформенный фреймворк, отечественные разработки, энергоэффективность.*

В погоне за снижением «порога входа» в разработку, на практике, с новыми технологиями и фреймворками такими как: Flutter, QML, React Native, Jetpack Compose и другими, мы получаем тотальное снижение уровня компетентности разработчиков. Зачастую разработчики остаются на том самом «пороге входа» и не развиваются дальше, система этого не предусматривает.

При снижении уровня компетентности, мы получаем приложения, которые всё более требовательны к железу и электроэнергии. Каждый новый фреймворк «высокого уровня» требует всё более производительных устройств для решения тех же задач. Это же касается и языков программирования «высокого уровня», таких как: Python, JavaScript, Dart [1].

Это же снижение компетентности ведёт к тому, что система больше не в состоянии делать по-настоящему сложные приложения, такие как: PLM, RM, PM, PDM, CAD, CAE, CAM, CAPP и тому подобные. Мы оказываемся завязаны на уже существующие бренды или, как минимум, уже существующие фреймворки.

Электроэнергия конечна. Особенно это заметно по требованиям для технологий с использованием искусственного интеллекта (далее ИИ), но и потребление ресурсов обычным железом тоже растёт. На сегодня, рост потребности в энергии не успевает за предложением, что легко заметить по росту тарифов во всём мире, включая Россию.

Задачи, которые пытаются решить с помощью ИИ, могли бы быть решены более энергоэффективно алгоритмически, будь для этого компетентные специалисты. Это касается, в первую очередь, задач статистического анализа, кластеризации, управления предприятием. Например, вместо использования ИИ-решения, как предлагает Дженсен Хуанг [2], можно использовать более простое решение на базе Steering behavior, экономя гигабаты вычислительных ресурсов.

Это также касается и представления данных. Вместо LLM имеет смысл использовать правильно организованную объектно-ориентированную БД, но правильно организовать её может только подготовленный специалист. ИИ здесь — затычка в дефиците кадров.

В отличие от специалиста, ИИ не развивается, не предлагает более глобальные решения, не создаёт собственные новые подходы. Для развития науки и техники костыль в виде ИИ — тупик, который выживает среднее звено прикладных специалистов. А без этого среднего звена у нас не будет высшего звена.

Без развития собственной школы программирования, от низшего до высшего уровня, однажды мы упрёмся в энергетический кризис ИИ, из которого не сможем выбраться.

С появлением на рынке китайских операционных систем (далее ОС) в дополнение к существующим российским разработкам, понадобятся специалисты, способные интегрировать их, включая уже существующие платформы, в общую систему. Без таких специалистов у нас будет два мира, отдельно — внешний, и отдельно — внутренние российские разработки, которые невозможно экспортировать.

В условиях замкнутости разработок внутри России, они не развиваются. Мы видим это на практике, например, по процессору Эльбрус и его поддержке новых технологий — не поддерживается транслятор

в LLVM IR. Рано или поздно, устаревание таких продуктов станет безнадежным, и они исчезнут.

Исчезновение отечественных разработок во многом ставит нас под контроль других государств. Даже если мы сами обеспечиваем критические системы, другие страны через мобильные ОС, браузеры, пользовательские приложения и решения вроде Firebase, фактически, собирают досье на наших гражданах, что даёт возможность их контролировать и провоцировать.

Для выхода из этой ситуации, отечественные продукты должны быть глобальными. А, значит, в том числе, поддерживать глобальные технологии, американские и китайские. Но, при этом, основываться на коде и системах под нашим контролем. Опять же, необходимы специалисты, которые способны этим заниматься системно, а не писать отдельные пользовательские приложения.

Отечественные разработчики ОС и другого большого софта не готовы поддерживать даже собственные версии зарубежных фреймворков [3]. Таким образом, единоразовое «портирование» быстро устареет, и обновлять софт, построенный на базе такого порта будет невозможным. Мы уже столкнулись с этим, например, для Qt. Без собственных квалифицированных разработчиков и собственного фреймворка эта проблема не решается.

Их можно понять, рабочее время специалиста дорогое, а специалистов мало. Ценовая конкуренция за специалистов активно развивается. Однако, эта конкуренция не решает проблему нехватки кадров, она её усугубляет, у людей с начальным уровнем подготовки формируются завышенные ожидания с одной стороны, и отпадает необходимость в саморазвитии, с другой.

Необходим системный подход к решению озвученных проблем. С одной стороны, необходимы собственные, отечественные решения, которые заместят бы проблемные зарубежные системы и фреймворки. С другой, необходимы специалисты, которые бы умели с ними работать, без таких специалистов и «популярности» в сообществе бизнес отказывается инвестировать в подобные проекты и смотрит на них скептически.

Например, создание образовательно-практического консорциума, который одновременно бы и готовил специалистов для компаний по отечественным технологиям, и занимался развитием этих технологий на благо всех участников консорциума.

В рамках такого консорциума, наладить образование не «для конкретного фреймворка» или «для конкретной ОС», а методологически цельную подготовку, в совокупности дающую уровень инженера-программиста.

В качестве технологической основы для такой образовательной программы мы предлагаем использовать открытый отечественный кроссплатформенный фреймворк Stappler SDK. Фреймворк прост в освоении и подходит для обучения новых разработчиков широкого профиля, так как в основном использует стандартные подходы, принятые в индустрии. Данный проект ориентирован на максимальный охват платформ, в том числе Эльбрус, ARM (Байкал), RISC-V, потенциальные китайские и индийские процессоры. Поэтому в качестве основного языка используется C++, и предусмотрена возможность использовать другие языки через виртуальную машину WebAssembly. Для получения наиболее простого и эффективного API, во фреймворке объединяется классический подход к разработке на «старом» языке C++ с практиками, пришедшими из других систем и платформ. Заимствуются техники из Rust, Golang, Java, и, одновременно, практики программирования системного уровня вроде пулов памяти и невладеющих контейнеров. Фреймворк предназначен для разработки и обслуживания разнородных информационных систем, таких как: серверные приложения, мобильные и настольные приложения, автоматизированные рабочие места, средства автоматизации, средства тестирования и многие другие. Кроме того, Stappler SDK полностью адаптирован для различных отечественных ОС.

Литература

- [1] Rui Pereira, Marco Couto, Francisco Ribeiro, Rui Rua, Jácome Cunha, João Paulo Fernandes, and João Saraiva, *Ranking Programming Languages by Energy Efficiency*, 2021
- [2] *CES 2025: AI Advancing at 'Incredible Pace,' NVIDIA CEO Says*, [Электронный ресурс] — URL: <https://blogs.nvidia.com/blog/ces-2025-jensen-huang/> (дата обращения 16.01.2025).
- [3] *Как мы раскрыли внутреннюю архитектуру Flutter и затащили его на собственную платформу*, [Электронный ресурс] — URL: <https://habr.com/p/864200/> (дата обращения 16.01.2025).

Александр Речицкий, Алексей Брагин

Ставрополь, Москва, АНО ДПО Академия TOP, МГТУ им. Баумана

Проект: ReactOS <https://reactos.org>

Как студенту получить первый реальный опыт работы в крупном проекте на примере ReactOS?

Аннотация

В докладе рассматриваются шаги, которые помогут студенту получить первый реальный опыт работы в крупном open-source проекте на примере ReactOS. Описаны ключевые этапы: изучение проекта и его документации, выбор подходящей задачи, взаимодействие с сообществом разработчиков и отправка первого кода. ReactOS — это свободная операционная система с открытым исходным кодом, основанная на архитектуре Windows, стремящаяся к высокой совместимости с существующими драйверами устройств и программами для ядра Windows NT 5.2 и в перспективе его более новыми версиями.

Ключевые слова: *reactos, первый опыт работы, портфолио, open source.*

Отсутствие реального опыта работы в крупных проектах является одной из ключевых проблем студентов IT-специальностей. Проект ReactOS предоставляет широкие возможности для преодоления этого барьера, предлагая доступ к крупной кодовой базе, поддержку сообщества и участие в профессиональных мероприятиях. ReactOS — это открытая операционная система, совместимая с Windows, что делает её ценным инструментом для изучения и практического применения в сфере системного программирования.

Социализация в рамках проекта играет важнейшую роль. ReactOS предоставляет возможности для взаимодействия как в онлайн-пространстве, так и в реальном мире. Через официальные каналы общения (форумы, чаты и GitHub-репозиторий) студенты могут общаться с опытными разработчиками, получать обратную связь и участвовать в обсуждениях. Важной частью жизни проекта являются мероприятия: участие в выставках, форумах разработчиков и собственный хакатон ReactOS Hackfest. Эти события позволяют участникам не только развивать навыки, но и лично познакомиться с командой проекта, создавая ценные профессиональные связи.

Для успешной интеграции студенту необходимо начать с изучения архитектуры ReactOS, настройки среды разработки и знакомства

с ключевыми участниками сообщества разработчиков. После подготовки и первых взаимодействий с командой студент может приступить к выполнению простых задач. В проекте ReactOS это может быть доработка файлов языковой локализации интерфейса, исправление мелких ошибок, улучшение документации или доработка уже существующего функционала. Подача Pull Request и получение обратной связи от опытных разработчиков помогают улучшить навыки программирования и понять стандарты крупного проекта. Социальная активность, включающая обсуждения и участие в мероприятиях, укрепляет связи с сообществом и повышает доверие к новому участнику.

Особенностью ReactOS является возможность быстрого карьерного и профессионального роста внутри проекта. Зарекомендовав себя с хорошей стороны, участник может уже через несколько месяцев претендовать на оплачиваемый контракт разработчика, финансируемый из пожертвований, либо подать заявку на участие в международных программах оплачиваемых стажировок, таких как Google Summer of Code. Эти возможности делают участие в ReactOS не только способом получения опыта, но и серьёзным шагом в профессиональной карьере.

Литература

- [1] Брагин А. В., Иванов И. П. *Операционные системы. Лабораторные работы* : учеб. издание. — Москва : Изд. МГТУ им. Н. Э. Баумана, 2024. с. 36.

Михаил Савин
Москва, МПГУ

«Умная игрушка» руками студентов-гуманитариев

Аннотация

Современные средства разработки электронных устройств для детей и школьников включают в себя в том числе открытые программы и аппаратные платформы. Мы ищем подходы и пытаемся разобраться в их возможностях и пригодности для использования студентами педагогического ВУЗа, не имеющими технического профиля подготовки. Экспертное мнение педагога при проектировании любой «умной

игрушки» может быть сформулировано более чётко и осознанно, если он будет иметь ясное представление о схемотехнике, архитектуре и логике программирования современных цифровых устройств. Доклад рассказывает в том числе и о трудностях, возникающих при использовании СПО в этом направлении. Ключевые слова: DSL (предметно-ориентированные языки), визуальные языки программирования, Ардуино, FBD (языки функциональных блоковых диаграмм)

Ключевые слова: *DSL (предметно-ориентированные языки), визуальные языки программирования, Ардуино, FBD (языки функциональных блоковых диаграмм).*

В этом году мы будем отмечать 40 лет, как в нашей стране у школьников появился предмет Информатика. Разработанный курс и учебник под руководством академика Андрея Петровича Ершова позволил включить его в основную программу обучения, а не факультативно. Темы строения компьютеров, алгоритмизации, программирования с тех пор стали интересны учащимся независимо от их будущих специальностей.

Современный микропроцессорный мир диктует нам условия, когда уже не «персональный компьютер» на столе, а гаджет в кармане (или на руке) является устройством с которым человек проводит большую часть времени. Но компьютеризация и цифровизация не стоят на месте. И очень сложно предугадать какими будут устройства будущего. Футуролог и преподаватель МТИ Дэвид Роуз в своей книге [1] предполагает, что дальнейшее «растворение» микропроцессорной базы в «умных вещах» в будущем неизбежно.

Если направление развития «hard» по-видимому это роботизация. То «soft», возможно, будет развиваться в сторону всё больших систем с элементами Искусственного Интеллекта (ИИ). Уже делаются прогнозы по которым не только прикладные программы, но и операционные системы могут быть сгенерированы ИИ «на лету», под конкретную задачу пользователя. Так ли необходимо в этом будущем педагогу (не математических и не естественно-научных дисциплин) глубоко разбираться в строении компьютеров и их программировании? На этот непростой вопрос нельзя дать краткого ответа. Поэтому мы в своём исследовании переформулируем его иначе: «Необходимы ли знания элементов математической логики, основ алгоритмизации, программирования, строения и архитектуры вычислительных систем и т.п. для экспертного мнения педагога будущего, при создании обучающихся и развивающих электронных устройств?» Очевидно, что уже

этот вопрос имеет однозначный положительный ответ. И, не смотря на развитие ИИ, преподаватель будущего должен быть компетентен в строении и принципах работы техники.

Исходя из этого предположения, мы выбираем направление от продукта к инструментам, необходимым для разработки устройств. Сам продукт может иметь разнообразное представление: обучающая компьютерная игра, обучающий и/или тренирующий навык чат-бот, «умная игрушка» и т.п. Рассматривая строение «умных устройств» мы способны изучить внутреннее строение процессоров(контроллеров) и их работу.

Если для изучения алгоритмов и математической логики вполне достаточно визуальных Scratch, CodeBlocks, ArduBlock и подобных, то для понимания архитектуры и потоков данных удобнее использовать графические предметно-ориентированные языки (DSL), к примеру FBD — язык блоковых диаграмм.

Нам известно два достаточно полноценных проекта, которые позволяют использовать язык FBD для программирования микроконтроллеров.

Первый — OpenPLC [2] является открытым и свободным ПО с обширным функционалом. Из его недостатков можно выделить отсутствие русскоязычной документации и окружения. А так же некоторую сложность в установке и использовании для неподготовленного пользователя. Достоинством же является возможность создавать программы для Arduino, RaspberryPi, ESP32 и др.

Второй рассматриваемый пакет — FLProg [3], к сожалению, не является ПО с открытым исходным кодом. Но его несомненное преимущество — он создан русскоязычным программистом, что позволяет снизить порог вхождения. А видеоролики самого автора дают возможность начать программировать микроконтроллеры сразу после установки среды.

К сожалению, нельзя не отметить и множество недостатков у FLProg. Размер пакета очень велик, и его установка занимает продолжительное время. В версии для Linux допущена досадная ошибка — использование только определённых шрифтов, которые, скорее всего, были установлены у автора программы во время компиляции. Из-за этого приходится вручную экспериментировать со шрифтами и их названиями в XWindows. Так же стоит отметить досадные ошибки в документации, которые не исправляются автором многие годы. Несмотря на эти недостатки, FLProg, видимо, остаётся единственной

альтернативой для изучения программирования микроконтроллеров средствами FBD.

Мы на занятиях по информатике исследовали различные варианты для обучения студентов работы с микропроцессорной техникой. К примеру, для студентов музыкального факультета было очень интересно создавать аналог терменвокса, в основе которого лежали ультразвуковые датчики. Или другой пример — роботдвигающийся по командам из мессенджера Telegram, спроектированный студентами факультета начального образования.

Хотя аналоги визуального Scratch являются отличными инструментами для начального обучения, FBD предлагает более сложный и глубокий подход, который может быть особенно привлекательным для студентов-гуманитариев, стремящихся развить свои навыки и применить их в реальных проектах. Этот подход не только развивает технические навыки, но и способствует междисциплинарному мышлению, что является важным аспектом в гуманитарных науках. Развитие же свободных трансляторов для этого языка могут способствовать большему интересу к СПО.

Литература

- [1] Роуз, Дэвид, Будущее вещей: Как сказка и фантастика становятся реальностью, 2015
- [2] OpenPLC, <https://autonomylogic.com/>
- [3] FLProg, <https://flprog.ru/>

Давид Султаниязов
Санкт-Петербург, ООО «РУС.ЯЗ»
<https://корсаков.рус>

Корса́ков: человек и российский язык программирования

Аннотация

Доклад посвящён студенческому научному проекту по разработке отечественного языка программирования с использованием свободного программного обеспечения, не зависящего от иностранного влияния на развитие языка, поддерживаемого на процессорах архитектуры x86_64, ARM¹, e2k¹, Loongsoon¹, RISC-V¹ и других. Целью доклада является ознакомление участников конференции с проектом и освещение темы грантовых конкурсов для получения студентами финансирования на развитие социально значимых проектов.

Ключевые слова: *язык программирования, отечественное программное обеспечение, язык общего назначения, проектная работа*

Корса́ков — язык, на котором хочется говорить

Корса́ков — язык программирования, в основе которого лежит идея использования кириллической раскладки и независимости от других языков высокого уровня. Назван в честь Семёна Николаевича Корса́кова — врача, изобретателя механических устройств. Он одним из первых представил идею «интеллектуальной машины» в начале XIX века, раньше Бэббидж, однако его труды не приняли в Российской академии наук, из-за чего он решил публиковаться во Франции и был малоизвестен общественности.

У программиста нет дедлайнов, есть только задачи

Задачи проекта:

- Популяризация программирования среди широкой аудитории и привлечение новых разработчиков на уровне образовательной системы

¹В планах.

- Обеспечение процесса обучения новых специалистов, что позволит привлекать в ИТ-сферу большее количество квалифицированных кадров
- Сохранение суверенитета как на уровне кода, так и на уровне мышления разработчиков
- Обеспечение независимости в области информационно-коммуникационных технологий
- Повышение уровня защищённости цифровых ресурсов

У нас же уже есть своё...

Корсаков призван повысить прозрачность работы с программным кодом для носителей русского языка. На данный момент аналогов отечественному мультипарадигменному кириллическому кроссплатформенному языку программирования общего назначения на рынке нет.

Близкими аналогами можно назвать:

- КуМир — язык, направленный на обучение азам программирования — синтаксически является кириллической интерпретацией языков BASIC и Pascal
- 1С:Предприятие — язык, направленный на обработку отчётов и взаимодействие с документооборотом

Мы считаем, что кириллический язык программирования позволит существенно снизить барьеры в изучении информатики, так как позволит реализовать образовательный процесс исключительно на родном языке, что позволит привлечь большее количество русскоговорящих студентов и учеников. Благодаря общей направленности, Корсаков может быть использован в качестве инструмента для решения задач в любых отраслях. Так как язык является полностью нашей разработкой — мы можем адаптировать его под нужды российского рынка.

Источник идей — люди

Разработка Корсакова является моей научной работой, которая началась с разговора с преподавателями на тему сохранения суверенитета страны и возможности разработки своей IDE для замены

недоступных в нашей стране программных решений. Планирование подобного проекта заняло примерно полгода, и начиная с февраля 2023 года была начата разработка прототипа на языке Python. Сейчас уже активно ведётся разработка обособленной версии компилятора — вернее, транслятора, который приводит высокоуровневый код в язык Ассемблера (диалект FASM). В следующих версиях код будет компилироваться напрямую в исполняемый файл.

Государство одобряет!

Подобный проект невозможно реализовать в полной мере без больших вложений как финансовых, так и человеческих, поэтому было принято решение подать заявку на грантовый конкурс Фонда содействия инновациям. Для этого потребовалось провести отдельную работу, о которой также будет рассказано в докладе: от сбора данных о рынке до формирования финальной заявки.

Релиз проекта планируется на февраль 2025 года.

«Корсаков». Снова по-настоящему твой! Пишите по-русски!

Литература

- [1] Ссылка на репозиторий Python-прототипа: https://gitverse.ru/rus.yaz/korsakov_python
- [2] Ссылка на сайт проекта: <https://корсаков.рус>

Алексей Драгунов, Алексей Нургалиев, Павел Алов,
Сергей Еремин

Псков, ГБУ Псковской области «Региональный центр информационных технологий»

Проект: Open Educational Environment
<https://git.integratics.ru/open-educational-environment>

**Создание облака для сферы образования на базе
свободного программного обеспечения**

В статье рассматриваются возможности создания облака для сферы образования, интеграции развёрнутых на базе региональной серверной инфраструктуры сервисов, построенных на базе свободного программного обеспечения; указываются основные проблемы, возможные направления исследований и разработок. Авторы рассказывают о дальнейшей реализации проекта «Открытая образовательная среда», который был представлен на конференции в 2024 году [1].

Ключевые слова: *nextcloud, keycloak, onlyoffice, MediaCMS, OpenID, образование, интеграция.*

Введение

Исторически сложилось, что система образования России была вовлечена в активное использование западных облачных сервисов, которые активно продвигались зарубежными компаниями, в первую очередь Microsoft, Intel и Google и до недавнего времени педагоги не могли представить жизни без «гугла». Доступные и качественные западные интернет-сервисы покорили сердца учителей. С появлением санкций некоторые сервисы закрылись для России, некоторые стали предлагать урезанный функционал. Для ещё доступных сервисов существенно выросли риски прекращения функционирования в любой момент.

Основная часть

Есть, как минимум, два пути, позволяющих обеспечить качественные сервисы для образовательных организаций региона: аренда облачных сервисов у российских коммерческих провайдеров, таких, как VK, Yandex и др. и создание собственного облака для доступа к сервисам педагогов образовательных организаций и учащихся на арендованных у российских провайдеров ресурсах или на своих серверах. При этом надо отметить, что ряд возможностей для работы у педагогов есть в доступном бесплатном сервисе *myschool.edu.ru* и Сферум. Для организации полноценной работы в образовательной организации, нужны развитые сервисы для управления доступом к файлам, коллективной работы, а также дополнительные образовательные сервисы. Корпоративные сервисы есть в продуктах VK, Yandex и др., их стоимость оценивается около 300 рублей в месяц на пользователя.

В зависимости от опций, это минимально составляет для Псковской области около 15 млн рублей в год.

Благодаря национальному проекту «Образование» в рамках проекта «Цифровая образовательная среда» были приобретены сервера, имеющие следующие характеристики: 128Gb RAM, SSD 256, 2xSATA 12Tb, Xeon 2.3 GHz. Для построения «базового» облака мы используем 43 таких сервера. Стоимость серверов составила около 14 млн рублей. Таким образом, с учётом стоимости электричества, при выборе selfhosting-модели на базе opensource, выгода, начиная со второго года использования, составляет более 14 млн рублей ежегодно.

На серверах установлена операционная система Альт Сервер Виртуализации 10.2. Для развёртывания приложений, обеспечивающих сервисы, применяются docker-контейнеры. Для организации единого корпоративного хранилища с обеспечением избыточности и гибкого перераспределения дискового пространства, использован Ceph (<https://github.com/ceph/ceph>). Отдельным вопросом для нас был выбор подходов к квотированию дискового пространства, как сделать так, чтобы не занятое пока пространство, было доступно для общего использования и в тоже время была возможность управлять облаком на региональном уровне и в каждой образовательной организации? Для решения проблемы было принято решение о создании для каждой организации своего инстанса Nextcloud с настройкой взаимодействия между ними и единой аутентификацией через Keycloak. При этом решаются две задачи: централизованное управление конфигурациями Nextcloud и обеспечение аутентификации с использованием доступа к региональной государственной информационной системе «Цифровое образование Псковской области», которая в свою очередь интегрирована с ЕСИА (порталом Госуслуг).

Для решения первой задачи выполнено объединение отдельных docker-compose файлов Nextcloud, Keycloak и Onlyoffice в один yaml-файл. Были исключены конфликты наименований, портов, также потребовалось оптимизировать число контейнеров postgresql до одного, создавая необходимые базы данных и пользователей в нём. Проведена настройка сервисов для интеграции между собой, а также формирование дампов баз данных и docker volumes переменных. Это должно было обеспечить развёртывание сервисов в преднастроенном виде. Дополнительно потребовалось восстановление прав доступа и владельца файлов внутри docker контейнера nextcloud. Также возникла необходимость повторного включения плагинов через методы nextcloud для

того, чтобы сервис обнаружил наличие docker volumes переменных и плагины отобразились в веб интерфейсе и в настройках экземпляра nextcloud. Интеграция Nextcloud с Keycloak реализована с использованием плагина Nextcloud OpenID Connect user backend.

Для решения второй задачи разрабатывается java-плагин для keycloak, который обеспечит переадресацию запросов на сервер с Authserv, находящийся в защищённой сети. Authserv, в свою очередь, реализует вход через ЕСИА ЕПГУ. таким образом, педагоги, родители и учащиеся, работая с системами электронных журналов и дневников, электронного обучения и другими (см. <https://it.pskovedu.ru>), получают возможность использовать свободное программное обеспечение в открытом образовательном пространстве: Nextcloud с рядом стандартных плагинов, OnlyOffice, Draw.io. Дальнейшее развитие проекта предполагает включение в облако целого ряда сервисов в интеграции с NextCloud, включая MediaCMS (замена ютуб, с точки зрения собственной публикации видеоконтента педагогами и доступа к нему обучающихся без прерываний на рекламу и непрофильный контент (<https://github.com/mediacms-io/mediacms>), Mattermost (<https://github.com/mattermost/mattermost>) — система корпоративных коммуникаций. Дополнительно в настоящее время формируется перечень opensource-продуктов, которые будут включены в облако для системы образования — среди них научные и образовательные сервисы, lowcode-платформы, CRM/ERP которые помогут организовать образовательным организациям лучшее взаимодействие с родителями, совершенствовать хозяйственную деятельность и развивать внутреннюю автоматизацию. Задача решается в комплексе с развитием РИС «Цифровое образование Псковской области».

Литература

- [1] Драгунов А., Драгунов К., Гладченко Л. *Создание открытой образовательной среды на базе операционных систем Альт*. // IX Конференция «Свободное программное обеспечение в высшей школе». Тезисы докладов, URL: https://www.basealt.ru/fileadmin/user_upload/education-conf-2024/educonf24.pdf

Пётр Леляев, Ксения Быховская

Москва, Департамент информационных технологий Москвы

Проблемы и решения внедрения СПО в образовательной организации на примере общеобразовательной школы г. Москвы

Аннотация

В статье рассмотрены проблемы, возникшие в одной из московских школ при переходе на свободное программное обеспечение. Описаны взгляды на проблемы со стороны участников образовательного процесса и команды разработки, предложены способы их решения и проведена оценка перспектив перехода школ на СПО.

Ключевые слова: *школа, внедрение, СПО, обратная связь.*

Согласно Указу Президента Российской Федерации от 30 марта 2022 г. № 166 «О мерах по обеспечению технологической независимости и безопасности критической информационной инфраструктуры Российской Федерации» и Приказу Министерства цифрового развития, связи и массовых коммуникаций РФ от 18 января 2023 г. № 21 «Об утверждении Методических рекомендаций по переходу на использование российского программного обеспечения, в том числе на значимых объектах критической информационной инфраструктуры Российской Федерации, и о реализации мер, направленных на ускоренный переход органов государственной власти и организаций на использование российского программного обеспечения в Российской Федерации», с 1 января 2025 г. органам государственной власти и заказчикам запрещается использовать иностранное программное обеспечение на принадлежащих им значимых объектах критической информационной инфраструктуры [1].

Образовательные учреждения, в частности, школы, не относятся к критической инфраструктуре. Однако их перевод на СПО также необходим по ряду причин — начиная с того, что выпускники обязаны ориентироваться в импортозамещённом ПО, и заканчивая наличием вредоносного кода в проприетарных программах, в частности, обнаруженного командой разработки МОС. В рамках данной статьи авторы анализируют вопросы, связанные с уже осуществляющейся реализацией такого перехода в рамках образовательных учреждений г. Москвы на примере ГБОУ Школа № 192.

В настоящее время в школах г. Москвы используется такое учебное оборудование, как интерактивные панели, личные ноутбуки учителей, учебные компьютеры, моноблоки и планшеты, периферийное оборудование, а также специфическое оборудование для профильных классов — например, датчики, цифровые микроскопы, тренажёры, имитирующие поведение человеческого организма для медицинских классов и многие другие. К сожалению, для значительного количества такого оборудования не создано программного обеспечения для Unix-подобных систем, а для некоторых из них не написаны даже драйверы. Поэтому вопрос о полном переходе школ на СПО пока не имеет решения — очевидно, что либо необходима разработка программных решений производителями оборудования, либо замена данного оборудования на отечественные аналоги. Однако большинство устройств, используемых в повседневной работе сотрудниками и учениками московских школ, уже переведено на Linux — это МОС12, основанная на репозитории свободных пакетов ROSA.

В целом школьные учителя являются достаточно консервативными. Нельзя говорить о каком-то протесте против отечественных разработок — современная школа является крайне непростым местом работы, и на освоение любых новых технологий в нерабочее время просто не хватает ресурсов. В то же время многие проблемы решаются в том числе благодаря оперативной реакции вендоров [2], но необходимо отметить, что импортозамещение в образовании является и, вероятно, в течение ещё нескольких лет будет являться достаточно сложным процессом для всех его участников. Для поддержки сотрудников в московских образовательных организациях существует IT-служба (ранее это были просто системные администраторы), отвечающая сотрудникам на возникающие вопросы и помогающая решить проблемы. Также у МОС есть сообщество активных пользователей, помогающих друг другу — часто проблема, возникающая в одной школе, относится не только к ней, и однократное описание вопроса в чате или на вики-странице приводит к её решению у многих пользователей. Однако, несмотря на все принятые меры, в команду разработки МОС периодически приходят запросы, связанные с невозможностью специалистов IT-службы решить возникающие проблемы. В 2024 году один из таких запросов пришёл из ГБОУ Школа 192 и содержал описание нескольких случаев. Приводим их в интерпретации авторов, не являющихся представителями IT-специальностей:

- по интерактивной доске прошла муха, после этого на ней стало невозможно писать;
- проигрыватель открывается сразу в двух окнах — в одном изображение, в другом звук;
- на одном из устройств не определяются флешки;
- ученик нарисовал медузу, после этого доска повисла и виснет каждый раз при открытии этого рисунка.

Представители команды разработки МОС посетили указанное здание в сопровождении представителя IT-службы школы. После формализации задач их решения выглядели следующим образом:

- создание простой графической утилиты для регулировки чувствительности панели¹ [3];
- создание графической утилиты для очистки конфигов². Можно производить полную очистку, в таком случае конфиг программы определяется её разработчиками, или восстановление из `/etc/skel` (в случае его наличия) — такой конфиг хранится в конкретной операционной системе;
- отправка заявки в техподдержку на замену разъёма — он был физически повреждён.

Проблема с рисунком оказалась наиболее сложной и, вероятно, единственной, действительно требующей непосредственного вмешательства разработчиков команды МОС. В программе OpenBoard, по умолчанию использующейся в качестве доски для МОС12 на интерактивных панелях, нашлась уязвимость, приводящая к бесконечному циклу во время обработки определённого полигона, который был случайно нарисован учеником. После добавления условия к этому циклу проблема решилась. Отправка патча в апстрим не потребовалась, поскольку в следующей версии OpenBoard (1.7.0) разработчики сами обнаружили и устранили данную проблему, и патч перестал быть необходимым.

По итогам посещения школы команда разработки сформулировала следующий список проблем, возникающий при переводе образовательных учреждений на СПО:

¹<https://abf.io/import/xinput-calibrator-gui>

²<https://abf.io/import/configcleaner>

1. В некоторых случаях может присутствовать некомпетентность или ненадлежащее исполнение обязанностей ИТ-службой школы;
2. Отсутствие связи между ИТ-службой и командой разработки;
3. Отсутствие связи между пользователями и командой разработки;
4. Специфика поставленного в школы оборудования и существующего для него ПО.

Команда разработки МОС видит следующие способы решения описанных проблем:

1. В связи с курсом страны на полное импортозамещение в сфере информационных технологий крайне необходима подробная должностная инструкция для ИТ-службы в образовательном учреждении. В дальнейшем, когда на отечественное ПО перейдут все государственные учреждения, такая инструкция будет необходима и для каждого из них.
2. Очевидно, что у ИТ-специалиста должна быть связь с командой разработки на случай возникновения проблем, которые он не в состоянии решить самостоятельно. В то же время доводить ситуацию до прямой связи конечных пользователей с разработкой также не стоит. Такую связь можно осуществлять при помощи специально выделенного канала. В настоящее время такими являются чаты, организованные в мессенджерах Sferum и Telegram и не имеющие никакого официального статуса. В этом вопросе команда разработки надеется на нормативные акты, дополняющие приказы о переходе на СПО.
3. Пользователи также должны иметь возможность связаться с командой разработки напрямую, однако формат, очевидно, должен отличаться от описанного в данной статье. Авторы видят две возможности: в случае необходимости устранения ошибок в конкретных программах можно написать о них в треке задач, а в случае, когда пользователь имеет возможность доработки кода, должна быть возможность предложить патч. В основном исходные коды программ для МОС лежат на сервере `hub.mos.ru`, функционал которого предполагает обе этих опции, но пользователи должны знать о наличии таких возможностей — вероятно, в интерфейс некоторых программ необходимо добавить форму обратной связи.

4. Как уже было отмечено, полного решения данной проблемы пока нет. Однако для большого количества иностранного ПО давно существуют и активно разрабатываются свободные российские аналоги. В частности, ГИА по информатике в 9 и 11 классе уже несколько лет можно проводить с использованием МОС и софта для этой системы. Разработчики команды адаптируют ПО для нужд российского образования. Например, был создан патч, который производит поиск по тексту в документе в кодировке sr-1251 средствами операционной системы (в файловом менеджере Dolphin), что значительно упрощает, а главное, делает привычным для учеников решение задачи ГИА по анализу текста. Если говорить о более сложном ПО, то, например, существуют аналоги AutoCad (FreeCad), TrikStudio для робототехники, для медиа-классов есть нелинейный видеоредактор Kdenlive. Также активно создаются драйверы для периферийных устройств.

Посещая российские регионы, разработчики МОС делятся своим опытом с коллегами, работающими в системе образования, также получая обратную связь от использования системы. Кроме того, многие образовательные организации переходят на другие свободные дистрибутивы, получая достаточно времени для того, чтобы разобраться в принципах работы СПО до того, как переход станет строго обязательным. В целом можно сделать вывод, что при должном отношении всех участников процесса импортозамещения, а также понимания, что все решают одну задачу, в течение нескольких ближайших лет перевод образовательных организаций на свободное отечественное программное обеспечение будет успешно завершён.

Литература

- [1] Приказ Минцифры России № 21 «Об утверждении Методических рекомендаций по переходу на использование российского программного обеспечения, в том числе на значимых объектах критической информационной инфраструктуры Российской Федерации, и о реализации мер, направленных на ускоренный переход органов государственной власти и организаций на использование российского программного обеспечения в Российской Федерации». <https://digital.gov.ru/ru/documents/8755/>
- [2] Теряева, Н. Ю. *Школьные ноутбуки удалось вынуть из кладовки*, 2021 [электронный ресурс] URL: <https://d-russia.ru/shkolnye-noutbuki-udalos-vynut-iz-kladovki.html>

- [3] Неофициальная wiki сообщества IT в образовательных учреждениях. Категория: Доски, 2024 [электронный ресурс] URL: <https://it-help-school.ru/Категория:Доски>

Денис Зайка, Тамара Зайка

Донецк, ФГБОУ ВО «Донецкий государственный медицинский университет имени М. Горького» МЗ РФ

Использование СПО и технологической некромантии в обеспечении преподавания медико-биологических дисциплин

Аннотация

Применение тонких клиентов для обеспечения преподавания медико-биологических дисциплин в медицинском университете позволяет использовать морально устаревшую технику, предоставляя возможность при этом пользоваться современными программными средствами обучения и работы. Система виртуализации Proxmox VE, Linux Terminal Server Project и операционная система OpenWRT в медицинском университете в условиях экономии.

Ключевые слова: *медицинский университет, OpenWRT, LTSP, Proxmox VE.*

Преподавание в медицинском вузе, наверное, возможно без использования вычислительной техники, однако его в этом случае сложно назвать современным. В Донецком медуниверситете последние несколько лет множество дисциплин преподавались с использованием дистанционных технологий. Не исключение и преподавание медико-биологических дисциплин, кроме того система оценивания требует оценки студентов на каждом практическом или лабораторном занятии. Для обеспечения этой работы незаменимы автоматические формы проверки знаний, такие как компьютерное тестирование. Также очень большой объём учебного материала гораздо лучше усваивается студентами, если используются наглядные материалы, а современная электронная вычислительная техника (ЭВТ) способна воспроизводить любые такие материалы (звук, анимацию, видеоизображение), а также ЭВТ может использоваться для работы с моделями биологических процессов (даже интерактивным) и разнообразным биологическим программным обеспечением. Кроме этого развитие компьютерных сетей позволяет использовать различные информационные

источники, расположенные в Internet, и даже работать с дистанционными учебными курсами определённого профиля.

К сожалению, финансовые и организационные возможности в наших условиях не позволяют в достаточной степени обеспечивать учебные подразделения не только техникой и персоналом, необходимыми для использования ЭВТ в учебном процессе, но и техникой, необходимой для учебно-методической работы преподавателей.

Однако поставки дешёвой техники, а также подлежащей списанию из разных источников на протяжении последних лет сделали возможным оснастить несколько компьютерных классов в нашем вузе, а также полноценно использовать на кафедрах локальные вычислительные сети. Компьютерная техника, которую мы использовали, преимущественно устарела и неспособна эффективно работать с большинством современных операционных систем (ОС), но внедрение терминальных решений позволяет использовать её для решения большинства современных задач. Сейчас существует много свободных программных продуктов пригодных для внедрения терминальных решений — Thinstation, OpenThinClient, Linux terminal server project (LTSP), другие варианты, например своё лёгкое ядро FreeBSD/Linux из NFS.

После нескольких попыток работы с разными вариантами терминальных решений, мы решили внедрять Ubuntu/LTSP. Серверная ОС с LTSP в каждом случае была размещена в гипервизоре KVM, работавшем на Debian based ОС Proxmox VE начиная с версии 4.0, а до того на FreeBSD с virtualbox, что позволило разместить там ещё несколько виртуальных машин для решения различных задач (организация тестирования, разных Web ресурсов и т.д.). Клиенты не имеют дисковых накопителей (которые являются наиболее частыми причинами серьёзных отказов) и загружаются с помощью PXE, это снижает затраты на обслуживание техники и ПО. Классы используются не только в учебном процессе, но и в работе преподавателей.

Таким образом, всё это позволяет, несмотря на ограничения, добиться некоторого прогресса в преподавании предметов. Также такое решение даёт студентам возможность ознакомиться на практике со свободными программными продуктами, так как согласно результатам опросов большинство из студентов не использовали такое ПО раньше, а некоторые даже не знали о нём.

В организации локальных сетей также использовались преимущественно дешёвые устаревшие домашние маршрутизаторы. Критерия-

ми отбора была возможность использования с ними ОС OpenWRT. Так, например, на кафедрах фармакологии и медицинской биологии много лет успешно трудились морально устаревшие D-Link DIR-300 B1 с OpenWRT 15.05, которые обеспечивали многосегментную маршрутизацию в ЛВС, работу основного и резервного каналов в Интернет, тунелирование с шифрованием и другое. А также DIR-615 E4 с распаянным USB и оверлеем на USB флеш накопителе, который работал ещё и принтсервером. В настоящее время трудятся Beeline SmartBox Turbo+, которые оснащены достаточным объёмом оперативной памяти и NVRAM для выполнения почти всего круга возложенных на них задач.

Иван Туманов

Санкт-Петербург, ГБУ ДПО «Санкт-Петербургский центр оценки качества образования и информационных технологий»

Пример настройки APM ученика в ОС Альт

Аннотация

Удобная и беспроблемная эксплуатация отечественных ОС в компьютерном классе позволяет учителю не отвлекаться от преподавания непосредственно учебного материала.

Ключевые слова: *Home, wine.*

Постановка задачи

Одна из первых проблем, с которой сталкиваются все учителя в классах — исключение модификации рабочего окружения пользователя учениками или быстрый возврат их к начальным настройкам, тем более часто они работают под одной учётной записью. Механизмы ограничения прав на действия, киоск и встроенный гостевой сеанс оказались неэффективны.

Второй проблемой можно назвать то, что окружение для запуска Win-ПО вместе со всеми программами (например, КОМПАС-3D) является частью домашнего каталога (далее профиля) только одного пользователя, в многопользовательской среде потребуется установка ПО под каждым пользователем.

Из многолетнего опыта работы предлагается следующее решение.

Чистый ученик

Создаём пользователя `user0`, заходим под ним и настраиваем всё, что необходимо. Обычно это настройка внешнего вида, ярлыков программ, ссылок, запуск по одному разу всех основных приложений (для отключения окон первого запуска, языковых настроек и т.п.) и настройка браузера на безопасный поиск, стартовую страницу и закладки, нужные расширения (детский контент-фильтр, блокировка рекламы).

Далее надо указать системе использовать при входе пользователей профиль `user0` (если нет ранее созданного):

в файл `/etc/pam.d/system-auth-common` нужно добавить внизу строку

```
Session required pam_mkhomedir.so skel=/home/user0 umask=0077
```

Теперь каждый пользователь, у которого нет домашнего каталога (новый локальный или доменный) будет при входе копировать преднастроенный профиль `user0`.

Для автоматизации удаления профилей при перезагрузке (а также выключении/включении) системы необходимо в планировщике `cron` (под `root` командой `crontab -e` или редактируя файл `/var/spool/cron/root`) добавить строку для удаления профиля (например, `user1`)

```
@reboot rm -rf /home/user1
```

Можно указать несколько строк для нескольких учётных записей, можно использовать маски для удаления, в домене можно удалять каталог с доменными профилями.

После данных действий пользователь `user1` при входе будет идентичен `user0`, при перезагрузке все его данные удалятся, и он снова скопирует окружение с `user0`. Учитель может на перемене перезагружать весь класс при помощи ПО управления классом `Veyon` или `Eryptes`.

Для возможности хранения файлов можно или создать локально каталог с правами записи всем (например, `/srv/data`), на который сделать ссылку на рабочий стол пользователя `user0` или использовать сетевые каталоги

```
mkdir /srv/data
```

```
chmod o+rwX /srv/data
ln -s /srv/data/ /home/user0/Рабочий\ стол/data
```

Общий wine

Кроме проблемы «один пользователь, один wine» есть ещё проблема копирования профиля user0, как скелетного при входе новых пользователей. Wine создаёт каталог `.wine` в профиле пользователя, который может занимать несколько гигабайт.

Простого «вынесения» каталога `«.wine»` за пределы профиля как ссылки и назначения всем прав доступа недостаточно, wine проверяет именно владельца каталога `.wine` и 3х файлов реестра в нём. Для обхода данного действия надо вынести из user0 только каталог `drive_c`.

Сначала создадим wine-окружение (команда `winecfg` в консоли от пользователя user0). Далее уже выполним набор команд под root для переноса каталога `drive_c` в скрытый /srv, разрешение всем записи и создания на него символьной ссылки в user0:

```
mv /home/user0/.wine/drive_c /srv
chmod o+rwX /srv/drive_c -R
ln -s /srv/drive_c /home/user0/.wine/drive_c
```

Теперь wine будет общим и программы, установленные user0, смогут запускать другие пользователи, однако ещё необходимо, чтобы каждому пользователю в wine-каталоге создавался и win-профиль (в `C:\users`) с привязками к linux-каталогам пользователя (рабочий стол, документы,...), что требуется многим программам. Для этого надо в автозапуск (в конец файла `/home/user0/.bashrc`) добавить запуск команды `wineboot -u`, если ранее win-профиль не создавался:

```
if [ ! -d "/srv/drive_c/users/$USER" ]; then
    wineboot -u
fi
```

Также может понадобиться модификация ярлыков запуска wine-программ (файлы `.desktop` в подкаталогах `/home/user0/.local/share/applications/wine/Programs/`, чтобы убрать префиксы wine и привязку в путях пользователя user0. Параметры `Exec` и `Path` должны выглядеть следующим образом (на примере КОМПАС LT)

```
Path=~/.wine/dosdevices/c:/Program Files (x86)/ASCON/KOMPAS-3D LT  
V12/Bin/  
Exec=wine KOMPASLT.exe
```

Заключение

Теперь для изменения/установки программ надо зайти под `user0` и сделать нужные действия. Полезно делать резервные копии каталога `user0` и общего `wine`. Для уменьшения объёма `user0` можно в нём удалять каталог `.cache`.

В реальной практике в школе используется локальный пользователь с автологинем `luser`, который удаляется при перезагрузке. При необходимости входа в другую учётную запись надо выйти и зайти под нужным именем.

Все описанные операции воспроизводятся в графическом режиме.

Павел Бозин

Москва, МГУ

Проект: PyBabel <https://github.com/python-babel/babel/pull/1161>

Интеграция работы с памятью переводов в систему локализации PyBabel

Аннотация

Разработка расширенного функционала для инструмента PyBabel, позволяющего работать с памятью переводов. Разработка включает создание функции `rubabel concat`, предназначенной для объединения нескольких файлов в один, а также функции `rubabel merge`, обеспечивающей обновление файлов переводов с использованием памяти переводов. Подготовить `pull request` для внесения изменений в открытое сообщество разработчиков PyBabel, с соблюдением установленных ими дисциплин программирования. Предусматривается поддержка и сопровождение `pull request` в процессе его согласования и утверждения в сообществе.

Ключевые слова: *локализация, интернализация, разработка, rubabel, gettext.*

Современная разработка программ нуждается в эффективной локализации, что связано с управлением большими объёмами переведённых текстов. В нашем проекте мы добавили новые функции в PyBabel, чтобы упростить этот процесс.

Цель работы была в том, чтобы расширить возможности PyBabel с помощью новых функций `pybabel concat` и `pybabel merge`, которые являются аналогами `msgcat` и `msgmerge` из утилиты `gettext`. Это позволяет легче объединять и обновлять файлы переводов, что актуально для компаний, стремящихся избежать лишних затрат на сторонние инструменты управления переводами.

В рамках проекта была разработана функция `pybabel concat`, которая позволяет объединять несколько .po файлов в один. При возникновении конфликтов в переводах приоритет отдаётся переводу из первого файла, при этом флаги, местоположения и другая информация сообщений объединяются. Также был создан инструмент `pybabel merge`, который даёт возможность объединять файлы с использованием памяти переводов. Для работы с памятью переводов реализованы два режима: добавление перевода, только если он отсутствует в выходном файле, или перезапись существующих переводов из памяти.

Общий процесс перевода с новыми процессами состоит в следующем: сначала применяется команда `pybabel concat` для создания компендиума переводов, этот собранный компендиум затем редактируется определённой группой специалистов. Когда разработчик вносит изменения в код и добавляет новые тексты, он может переводить их самостоятельно, опираясь на компендиум как на основной или вспомогательный источник переводов. Схема этого процесса представлена на рисунке 1.

Новые функции облегчают управление переводами, предоставляя инструменты для автоматизации процессов «из коробки». Это выгодно для компаний, которые хотят минимизировать затраты на приобретение и внедрение сторонних инструментов для управления переводами. Даже в рамках учебных проектов добавленные функции окажутся полезными, поскольку они позволяют студентам не только работать с предоставленным преподавателем компендиумом переводов, но и использовать общие термины во всех проектах. Это упрощает понимание материала и уменьшает путаницу при работе с различными текстами, а также способствует более эффективному обучению в изучении процессов локализации.

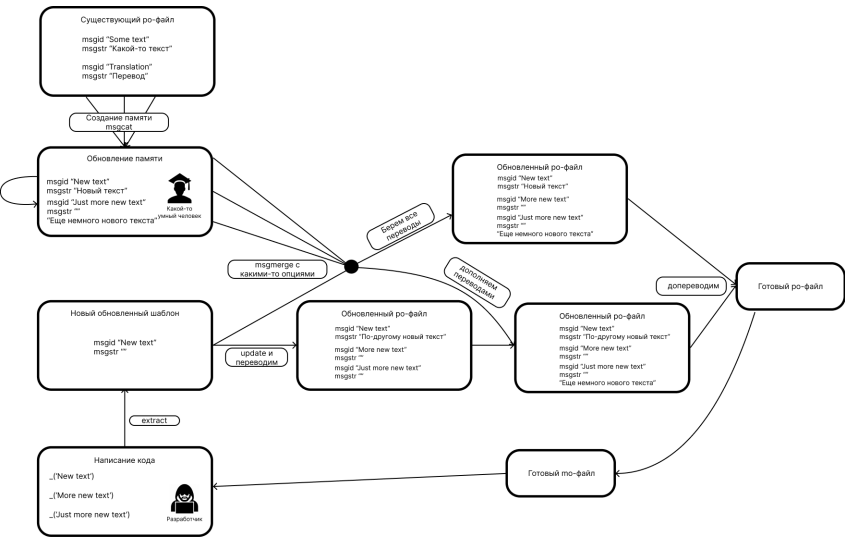


Рис. 1: Процесс перевода с использованием новых инструментов.

На момент написания тезисов коммит находится в состоянии pull request’a в репозитории сообщества PyBabel. Планируем довести процесс до конца, чтобы код официально был принят. Также планируется развитие этих инструментов и проверка их совместимости с дистрибутивом ОС ALT Linux. Проект выполняется в рамках курса на UNIX «Совместная разработка приложений на Python» и разработанные инструменты будут в него добавлены.

Полина Петруша, Иван Хахаев
Санкт-Петербург, СПбГЭТУ «ЛЭТИ»

Адаптация пользователей к реальности российского ПО

Аннотация

Обсуждается адаптация пользователей в условиях импортозамещения при массовом переходе на российские операционные системы, основанные на свободном ПО. Внимание уделяется особенностям пользовательского опыта, организационным и техническим решениям по обеспечению адаптации. Предлагается пошаговая стратегия реализации организационных и технических решений.

Ключевые слова: *импортозамещение, пользовательский опыт, свободное ПО, адаптация.*

Поскольку импортозамещение стало неизбежным в силу причин непреодолимого характера, возникает задача массового перехода на российские операционные системы, основанные на свободном ПО (на ОС семейства Linux). При решении этой задачи существует два основных барьера — совместимость периферийного оборудования и накопленный пользовательский опыт (UX).

UX можно поделить на нейтральный, необъяснимый и негативный.

К негативному UX относится:

- применение WinRAR для создания архивов;
- именование файлов длинными фразами на кириллице;
- устоявшаяся ассоциация между типами электронных документов и названиями приложений из MS Office.

К нейтральному UX можно отнести:

- использование клавиши «Win» для вызова меню приложений;
- использование «Alt+Shift» для изменения раскладок клавиатуры;
- двойной щелчок мышью для открытия каталогов и запуска приложений из ярлыка или по ассоциации;
- ожидание размещения панели задач в нижней части экрана с кнопкой вызова меню приложений в левой части панели задач.

К необъяснимому UX относится вера в единственно правильную организацию меню в приложениях из MS Office и других проприетарных программах (Adobe InDesign, CorelDraw! и т. п.), несмотря на изменения этого меню в зависимости от версии.

Из-за имеющегося UX после установки ОС семейства Linux на рабочие компьютеры пользователи испытывают негатив в связи с выходом из зоны комфорта в части пользовательского интерфейса (UI) и даже отказываются работать с такими ОС.

Понятно, что «в Linux можно настроить все», но опыт показывает, что в некоторых дистрибутивах после установки нужно потратить десятки минут для приведения системы в привычное для пользователя состояние.

Для ОС «Альт Образование» 10-й платформы при приведении UI в соответствие с UX дополнительно выполняются:

- установка шрифтов TrueType от MS;
- установка firefox-esr, добавление виджета запуска на панель задач;
- установка браузера от Яндекс, добавление виджета запуска на панель задач;
- отключение графических эффектов оформления рабочего стола;
- изменение шрифтов по умолчанию в LibreOffice на Arial и Times New Roman;
- добавление кнопок завершения сеанса, выключения и блокировки экрана на панель задач;
- изменение темы менеджера сеансов LightDM (для версии с XFCE).

В условиях конкуренции вендоров и дистрибутивов выигрывает вариант, который требует минимальных дополнительных настроек, вне зависимости от «технологичности» и объёма пакетной базы. Мы предлагаем пошаговую адаптацию.

Первый шаг — организация встречного движения:

- с одной стороны, обучение пользователей новому программному окружению;
- с другой стороны, адаптация этого окружения с учётом «нейтральных» привычек и доработок, специфичных для конкрет-

ной организации и ситуации (учебный класс, рабочее место сотрудника и пр.).

Рабочее окружение адаптируется по принципам:

- «одна задача — одно приложение»;
- именование приложений по назначению: сначала назначение, потом, в скобках, название конкретной реализации (например «Менеджер архивов (Ark)», а не «Ark (Архиватор)»).

Второй шаг: выработка и доведение до пользователей т. н. «корпоративного стандарта», устанавливающего основной набор приложений по их функциональному назначению, а также форматы файлов для обмена в пределах организации, основные требования к оформлению пользовательского окружения и шрифтам в документах.

Таким образом, делается кастомизация существующего базового дистрибутива в части оформления, набора приложений, стиля именования приложений, а также добавления на рабочий стол ссылок на основные корпоративные ресурсы. Выполнение такой кастомизации на каждом рабочем месте требует либо настроенного сервиса управления настройками систем (типа Ansible), либо (при отсутствии возможности создания таких сервисов для каких-то групп рабочих мест) создания кастомизированных установочных образов с оптимизированным составом приложений и настроек.

Для преподавателей вводится дополнительная мотивация — подготовка учебно-методических материалов под применение свободного ПО из состава российских ОС как показатель эффективного контракта.

На третьем шаге производится адаптация интерфейса управления системой для администраторов и специалистов.

На основе опыта использования дистрибутива «Альт Образование» 10-й платформы достаточно очевидными являются следующие доработки Центра управления системой (ЦУС):

- обеспечение мониторинга состояния процессов (в табличном и графическом виде);
- получение полной спецификации оборудования;
- доработка механизма отображения журналов служб (сервисов);
- разработка механизма добавления средств управления веб-серверами и серверами баз данных при их установке;

- синхронизация объектов управления в локальной и в web-версиях;
- переработка дизайна ЦУС (пример — YaST в OpenSUSE);
- реализация модуля для создания пакетов брендинга в диалоговом режиме;
- доработка модуля alterator-mkimage для обеспечения модификации образов ОС в диалоговом режиме.

Указанные задачи по доработке ЦУС могут быть реализованы в ходе выполнения ВКР студентов, а затем интегрированы в общую пакетную базу Sisyphus.

Андрей Филиппов

Архангельск, Северный (Арктический) федеральный университет
им. М. В. Ломоносова

Использование свободного программного обеспечения для анализа спектра радиочастот

Аннотация

Развитие свободного программного обеспечения позволяет решать различный спектр задач, дописывать, изменять и получать новые подходы и решения. Область телекоммуникационных систем достаточно специфична, но и в ней есть множество продуктов для различных систем, как правило они либо коммерческие, либо под операционные системы Windows. В Linux для решения задач радиосвязи применяется программное обеспечение GNU Radio, которое позволяет использовать различные решения и агрегировать работу устройств, а также модифицировать и развивать данное решение. В статье рассматривается программное обеспечение и решения для агрегации сетевого sniffера и передатчиков SDR.

Ключевые слова: *радиочастотный спектр, СПО, SDR, беспроводные технологии, анализ данных.*

Радиочастотный спектр представляет собой неисчерпаемый, но ограниченный природный ресурс, который играет ключевую роль в обеспечении работы современных беспроводных технологий и систем связи, таких как мобильные телефоны, Wi-Fi, спутниковая связь и интернет вещей, зависящих от его эффективного использования.

Традиционно исследование радиочастотного спектра проводилось с использованием дорогостоящего оборудования и специализированного программного обеспечения. Однако появление концепции программно-определяемого радио (SDR, Software-Defined Radio) открыло новые возможности. SDR представляет собой радиоприёмник и/или радиопередатчик, базовые параметры которого определяются программным обеспечением, а не аппаратной конфигурацией. В отличие от специализированных аппаратных платформ, SDR позволяет исследователям использовать универсальное оборудование в сочетании с гибкими программными решениями.

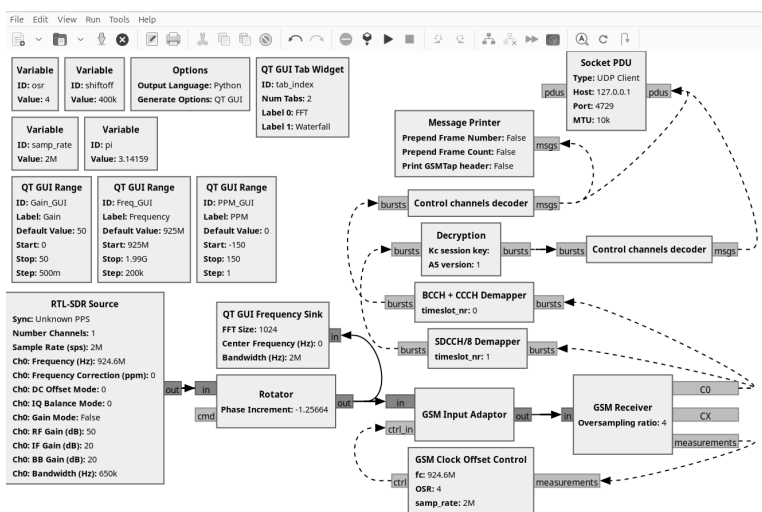


Рис. 1: Графический интерфейс GNU Radio

Свободное программное обеспечение (СПО), такое как GNU Radio, позволяет работать с различными сигналами, в том числе с использованием SDR. GNU Radio обеспечивает доступ к мощным инструментам анализа радиочастотного спектра и позволяет, собрав событийную схему путём соединения друг с другом различных блоков обработки сигнала, получить исполняемый код на языке Python. Та-

ким образом, посредством интеграций и агрегаций различных объектов, можно за короткое время получать прототипы, готовые к решению определённых задач. Примеры работы с GNU Radio показаны на Рис. 1 и 2.

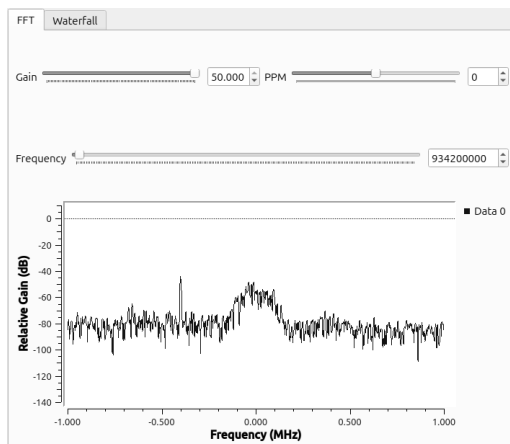


Рис. 2: График построенный с помощью GNU Radio

Цифровые системы передают информацию основываясь на множестве протоколов и принципов, описанных в стандартах, например RFC 768 для UDP или RFC 793 для TCP. Однако, в потоке трафика передаётся не только лишь полезная нагрузка, но и различная системная информация. Как раз такое СПО, как например Wireshark, посредством языка LUA позволяет описать структуру любого объекта, тем самым фильтруя из всего потока трафика лишь необходимые данные и классифицируя их. Пример работы Wireshark представлен на Рис. 3.

В качестве примера агрегации рассмотренного СПО представлен сниффер GSM. Приём сигналов осуществляется с использованием GNU Radio, в котором предварительно была собрана схема, после чего полученная информация передаётся в Wireshark, отфильтровывается и классифицируется. В результате работы программы пользователь может получать передаваемую полезную нагрузку в режиме реально-

No.	Time	Protocol	Info
20...	420.351...	GSMTAP (CCCH) (RR)	Paging Request Type
20...	420.357...	GSMTAP (CCCH) (RR)	Paging Request Type
20...	420.409...	GSMTAP (CCCH) (RR)	Paging Request Type
20...	420.426...	GSMTAP (CCCH) (RR)	System Information
<ul style="list-style-type: none"> Frame 8595: 81 bytes on wire (648 b) Ethernet II, Src: 00:00:00_00:00:00 Internet Protocol Version 4, Src: 1 User Datagram Protocol, Src Port: 5 GSM TAP Header, ARFCN: 0 (Downlink) SACCH L1 Header, Power Level: 5, T Link Access Procedure, Channel Dm (
<ul style="list-style-type: none"> GSM CCCH - System Information Type L2 Pseudo Length 0110 = Protocol discriminator Message Type: System Information T Cell Identity - CI (3172) Location Area Identification (LAI) Cell Options (SACCH) NCC Permitted SI 6 Rest Octets 			

Рис. 3: Пакеты трафика в программе Wireshark

го времени. Результат работы программы был представлен ранее на Рис. 3, фрагмент кода программы представлен ниже.

```
self.gsm_receiver_0 = gsm.receiver(4, [0], [], False)
self.gsm_message_printer_0 = gsm.message_printer(pmt.intern(''), False, False, False)
self.gsm_input_0 = gsm.gsm_input(
    ppm=PPM_GUI,
    osr=osr,
    fc=Freq_GUI,
    samp_rate_in=samp_rate,
)
self.gsm_decryption_0 = gsm.decryption([], 1)
self.gsm_control_channels_decoder_0_0 = gsm.control_channels_decoder()
self.gsm_control_channels_decoder_0 = gsm.control_channels_decoder()
self.gsm_clock_offset_control_0 = gsm.clock_offset_control(Freq_GUI-shiftoff, samp_rate, osr)
self.gsm_bcch_ccch_demapper_0 = gsm.gsm_bcch_ccch_demapper(
    timeslot_nr=0,
)
self.blocks_rotator_cc_0 = blocks.rotator_cc((-2*pi*shiftoff/samp_rate), False)
```

Полный код программы можно найти на GitHub: <https://github.com/EvilOny/GSM-Sniffer/tree/main>.

Представленный пример агрегации свободного программного обеспечения является лишь одним из множества вариантов. Потенциал подобных решений практически неограничен, так как можно принимать различные типы сигналов, выбирать из них необходимые данные, объединять и интегрировать разные инструменты. Это предоставляет возможность быстро создавать прототипы уникальных инструментов, адаптированных для решения конкретных задач, что делает такие подходы чрезвычайно гибкими и эффективными.

Литература

- [1] Тен, К. А., *Приборы и методы экспериментальной физики.*
- [2] Карась, А. В. *Радиоэлектронные устройства, используемые в полосе частот 2300–2400 МГц для систем телекоммуникаций* / А. В. Карась // Электронный сборник трудов молодых специалистов Полоцкого государственного университета / Полоцкий государственный университет ; ред. кол. : Д. Н. Лазовский (пред.) [и др.] . — Новополоцк : ПГУ, 2015. — Вып. 10 (80): Промышленность. — С. 35–36. URL: <https://elib.psu.by/handle/123456789/35452>
- [3] Яникеев, А. С., Жернаков, С. В., *Иновационные технологии.*
- [4] Володина, Е. Е., *Управление использованием радиочастот.*

Лев Харитонов, Антон Живечков

Арзамас, Арзамасский политехнический институт НГТУ им. Р. Е. Алексеева

Проект: `gnuplot-hpp` <https://github.com/LeoKhariton/gnuplot-hpp>

Разработка библиотеки для математической визуализации из консольных приложений на C++

Аннотация

В статье описывается разработка библиотеки для визуализации математических данных из консольных приложений на языке C++ с использованием `Gnuplot`. Библиотека предназначена для использования в рамках изучения дисциплин из области вычислительной математики.

Ключевые слова: *визуализация данных, математическая визуализация, вычислительная математика, библиотека, C++, Gnuplot.*

В образовательном контексте визуализация математических методов способствует более эффективному усвоению и интерпретации материала, предоставляя студентам интуитивно понятные и запоминающиеся представления сложных и абстрактных концепций. При изучении вычислительной математики студентам часто необходимо визуализировать методы и результаты своих вычислений для лучшего понимания того или иного численного алгоритма.

На данный момент существует множество инструментов для визуализации математических данных, начиная от простых онлайн-конструкторов графиков (*Desmos*, *GeoGebra*, и др.), и заканчивая крупными программными пакетами, предназначенными для научных вычислений (*GNU Octave*, *Scilab*, и др.).

Во многих университетах, в том числе и в Арзамасском политехническом институте, для изучения вычислительной математики и реализации алгоритмов численных методов традиционно используется язык C++. Было бы удобно отображать графики в рамках той же программы, где реализован и сам алгоритм. Хотя C++ является гибким, быстрым и универсальным языком, он не имеет встроенных средств для визуализации данных. Существующие решения и библиотеки могут быть сложными в установке и использовании или иметь достаточно примитивные возможности. Визуализация данных в отдельной программе зачастую увеличивает объём работы. Однако в рамках изучения вычислительной математики целесообразно, чтобы визуализация не становилась трудоёмкой и времязатратной задачей. В связи с вышеизложенным, тема работы является актуальной.

Целью данной работы является разработка библиотеки для визуализации данных из консольных приложений на C++.

Наиболее распространённые на данный момент графические пакеты и библиотеки для визуализации данных: *Gnuplot*, *Matplotlib*, *PLplot*, *Cairo*, *MathGL*, *NumRe*, *Plotly*, и др. Для решения данной задачи был выбран программный пакет *Gnuplot*, который представляет собой свободное кроссплатформенное ПО с командным интерфейсом. Несмотря на то, что первая версия *Gnuplot* вышла почти 40 лет назад, проект до сих пор активно поддерживается, обновляется и совершенствуется. *Gnuplot* поддерживает различные форматы ввода, включая команды и чтение из файлов, а также форматы вывода, такие как графическое окно, *L^AT_EX*, PDF, изображения в форматах PNG, SVG и т. д. [1, 2, 3].

Разработанная библиотека представлена в виде заголовочного файла `gnuplot.hpp`, содержащего класс *Gnuplot*. Этот класс включает методы, которые автоматически формируют и отправляют команды в *Gnuplot* на основе входных данных. Таким образом, команды *Gnuplot* инкапсулированы в методы C++.

Основные функции библиотеки включают: `plot` и `plot3d` для построения двумерных и трёхмерных графиков соответственно по заданным массивам координат или строке-формуле, `plot_dot` для ри-

сования точек на графике, `plot_xerr`, `plot_yerr`, `plot_xyerr` для построения графиков с погрешностями по осям X и Y , `plot_vectors` для построения векторных полей, `histogram` для построения столбчатых гистограмм, `multiplot` для отображения нескольких графиков на одном рисунке, `show` для отображения добавленных графиков, `save_as_png` и `save_as_svg` для сохранения графиков в форматах PNG и SVG соответственно, а также методы для установки заголовков и диапазонов осей, такие как `set_title`, `set_xlabel`, `set_ylabel`, `set_zlabel`, `set_xrange`, `set_yrange`, `set_zrange` и `set_grid` для включения отображения сетки на графике.

Для использования библиотеки необходимо установить Gnuplot на компьютер и скопировать заголовочный файл библиотеки в проект.

На Рис. 1 на стр. 129 представлены графики и диаграммы, демонстрирующие основные функциональные возможности библиотеки: графики в декартовых и полярных координатах, 2D- и 3D-пространстве, графики с погрешностями, гистограмм, векторных полей и т. д.

В качестве примера, визуализируем решение следующей системы линейных алгебраических уравнений (СЛАУ):

$$\begin{cases} 2x + 0,3y + 0,5z = -0,7 \\ 0,1x + 3y + 0,4z = 3,7 \\ 0,1x + 0,1y + 4z = 8 \end{cases}$$

Минимальный C++-код, необходимый для решения этой задачи с использованием разработанной библиотеки, представлен в Листинге ниже.

Минимальный код для визуализации решения СЛАУ

```
#include "gnuplot.hpp"

int main() {
    Gnuplot plt { };
    plt.set_title("Система линейных уравнений");
    plt.set_xlabel("x");
    plt.set_ylabel("y");
    plt.set_zlabel("z");
    plt.plot3d("(-0.7-2*x-0.3*y)/0.5", "2x + 0,3y + 0,5z = -0,7");
    plt.plot3d("(3.7-0.1*x-3*y)/0.4", "0,1x + 3y + 0,4z = 3,7");
    plt.plot3d("(8-0.1*x-0.1*y)/4", "0,1x + 0,1y + 4z = 8");
}
```

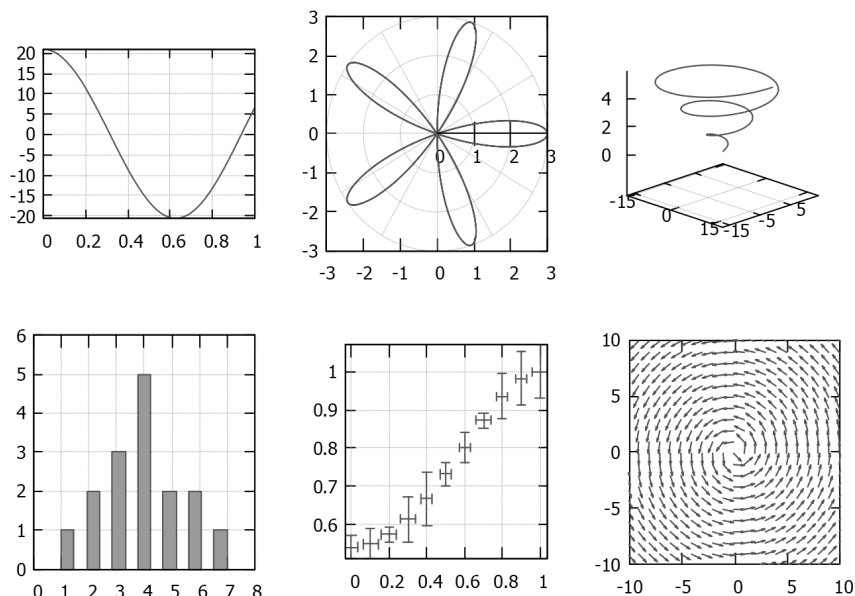



Рис. 1: Графики, выполненные с помощью разработанной библиотеки

```
plt.plot_dot({ -1, 1, 2 }, "Решение (-1, 1, 2)");
plt.set_grid();
return 0;
}
```

Результат работы программы представлен на Рис. 2 на стр. 130. Поскольку решалась система из трёх уравнений с тремя неизвестными, на графике изображены три плоскости, пересекающиеся в точке решения.

Разработанная библиотека успешно прошла апробацию в Арзамасском политехническом институте со студентами, обучающимися по направлениям подготовки «Прикладная математика» и «Информационные системы и технологии» в рамках лабораторных работ по дисциплинам «Численные методы» и «Вычислительная математика» соответственно.

Таким образом, была разработана библиотека для визуализации математических данных из консольных приложений на языке C++.

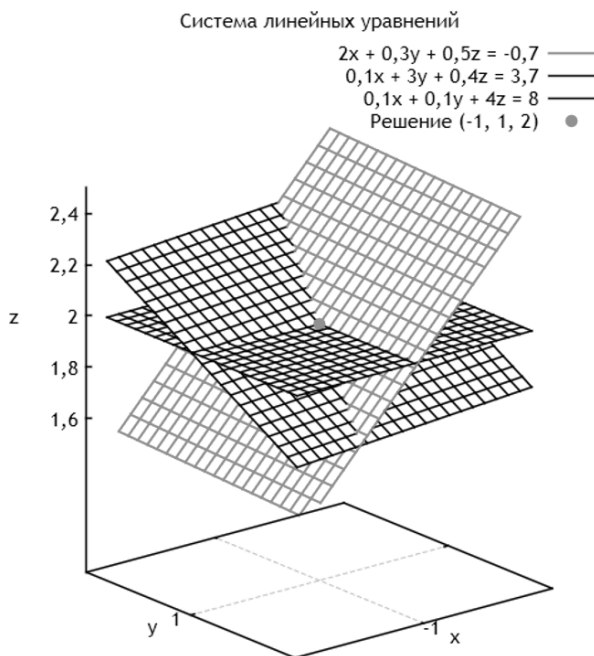


Рис. 2: Визуализация решения СЛАУ

Она предоставляет удобный и интуитивно понятный интерфейс для построения графиков с использованием **Gnuplot**, аналогично тому, как это делается при помощи **Matplotlib** или **GNU Octave**. В будущем планируется расширение функционала, включая поддержку анимации, а также дальнейшее улучшение существующих методов.

Литература

- [1] Janert P. K. *Gnuplot in action: understanding data with graphs*, 2nd edition. Manning, 2016. 400 p.
- [2] Попов А. В. *Gnuplot и его приложения*. М.: Издательство попечительского совета механико-математического факультета МГУ, 2015. 240 с.
- [3] Williams T., Kelley C. *gnuplot 6.0. An Interactive Plotting Program*. 2023. URL: <https://gnuplot.sourceforge.net/docs/gnuplot6.html>.

Александр Кузьмин

Санкт-Петербург, Военная академия связи им. С. М. Будённого

Проект: SpeedAPI <https://github.com/WolfMTK/speedy>

Сложности разработки веб-фреймворка на языке программирования Python

Аннотация

В данной статье рассматриваются основные сложности разработки веб-фреймворка на языке программирования Python. Цель работы заключается в подробном описании тех проблем, с которыми возможно столкнуться при разработке открытого проекта.

Ключевые слова: *python, FastAPI, инверсия управления, фреймворк, внедрение зависимостей.*

Введение

Разработка библиотек, фреймворков являются не самыми простыми задачами, которые могут возникнуть у многих разработчиков. Как многие компании пытаются экспериментировать со своей продукцией: добавлять новый функционал для своих клиентов или же сотрудников; подстраивать свой продукт под бизнес; экспериментировать на тестовых стендах и развивать инфраструктуру внутри компании, так и разработчикам в какой-то момент приходится разрабатывать свою библиотеку, которая бы приносила пользу не только бизнесу, но и тем, кто занимается экспериментами в разработке.

Различие между фреймворком и библиотекой

Библиотека — это набор подпрограмм или определённых классов. В библиотеке код можно использовать повторно, то есть тот код, который написан другими разработчиками. Классы и подпрограммы обычно определяют конкретные операции в определённой области. Например, существуют математические библиотеки, которые позволяют разработчику вызывать функцию, не повторяя реализацию алгоритма [1].

Фреймворк, в отличие от библиотеки, обычно намного сложнее. Он определяет «скелет», в котором приложение определяет свои собственные функции для создания некоторой архитектуры проекта. Таким образом, код будет вызываться самим фреймворком при необходимости. Преимуществом является то, что разработчику не нужно беспокоиться о том, хорош ли дизайн или нет, а нужно только реализовывать функционал, характерный для конкретной области [1].

Самое важное различие и, по сути, определяющие различие между библиотекой и фреймворком — это инверсия управления.

Инверсия управления

Инверсия управления — распространённое явление, с которым вы сталкиваетесь при расширении фреймворков. На самом деле, это часто считается определяющей характеристикой фреймворка [2].

Инверсия управления — это ключевая особенность, которая отличает фреймворк от библиотеки. Фреймворк представляет собой некий абстрактный дизайн с наиболее сложным поведением.

Внедрение зависимостей

Внедрение зависимостей — одна из наиболее превратно понятных концепций в объектно-ориентированном программировании. Путаница в этом вопросе весьма велика и распространяется как на терминологию, так и назначение и механизмы внедрения зависимостей.

Внедрение зависимостей — это набор связанных принципов и паттернов. Оно является в большей степени процессом обдумывания и проектирования кода, а не конкретной технологией. Основная цель использования внедрения зависимостей — создание удобного в сопровождении программного обеспечения, построенного на объектно-ориентированной парадигме [3].

Веб-фреймворк SpeedAPI

SpeedAPI — это открытый проект, находящийся в стадии разработки. Он является аналогом таких веб-фреймворков на Python, как FastAPI, Litestar и Starlette. Проект частично реализует спецификацию ASGI, а в будущем планируется добавить встроенный сервер для

разработки, подобный тому, что предлагается «из коробки» в Django и Flask.

Разработка собственного веб-фреймворка — это сложная задача, требующая решения сразу нескольких непростых вопросов. Самое важное — создать архитектуру, которая станет основой для дальнейшего развития проекта. Помимо этого, нужно разобраться со спецификациями, изучить опыт других фреймворков и выбрать подходящие библиотеки. Но сложность заключается не только в технической стороне. Чтобы проект жил и развивался, важно привлечь внимание сообщества, найти единомышленников, которые помогут с разработкой, и донести до них идею. Это требует не меньше усилий, чем написание кода.

Развитие веб-фреймворка SpeedAPI

В будущем в веб-фреймворке SpeedAPI планируется добавить встроенный сервер разработки, который позволит проверять работоспособность приложения без необходимости использования сторонних серверов, таких как Unicorn или Nginx. Также планируется реализация технологического слоя, который обеспечит взаимодействие между компонентами и маршрутизацию. Среди других задач — добавление автоматической генерации спецификаций OpenAPI и внедрение концепции управления зависимостями, что сделает разработку приложений более удобной и гибкой.

Сложности разработки и сопровождения проекта

Многие разработчики, которые только начинают работать с открытым программным обеспечением, часто задаются вопросом: «С чего начать, чтобы быть полезным проекту, и как включиться в процесс разработки?». В большинстве зрелых открытых проектов есть понятная «дорожная карта» — список задач, таких как исправление ошибок или улучшение безопасности. С ней можно ознакомиться, изучая проект, который привлёк внимание, и выбрать подходящую задачу для старта. Сложности возникают, если проект находится на ранней стадии развития. Часто у разработчиков есть множество идей о том, каким должен быть функционал, но реализация тормозится, когда проект ведёт один человек или разработка приостанавливается из-за недостатка опыта у команды.

Литература

- [1] Durai Amuthan H, *What is the difference between a framework and a library*, <https://stackoverflow.com/a/15600924>
- [2] Martin Fowler, *Inversion Of Control*, <https://martinfowler.com/bliki/InversionOfControl.html>
- [3] Симан М., *Внедрение зависимостей в .NET*. // — СПб.: Питер, 2014. — 464 с.,

Ольга Семёнова

Москва, РГУ нефти и газа (НИУ) имени И. М. Губкина

Проект: AU_Team <https://github.com/auteam-usr/science/ansible-alt>

Обзор инструментов статического анализа уязвимостей в плейбуках Ansible

Аннотация

Работа посвящена изучению подхода Infrastructure as Code (IaC) и его реализации с использованием инструмента автоматизации Ansible. Рассмотрены принципы работы Ansible, включая использование плейбуков, а также выявлены проблемы безопасности. В рамках исследования проведён статический анализ конфигурации для управления полнодисковым шифрованием LUKS в ОС Альт.

Ключевые слова: *Статический анализ, уязвимости, Ansible.*

Подготовка современных специалистов, обучающихся в ВО по УГСН 10.00.00, строится на изучении актуальных технологических решений, развивающих отечественные технологии. В Доктрине информационной безопасности определено, что подготовка специалистов является основополагающим направлением обеспечения национальной безопасности Российской Федерации [1]. Разработка современных методик и стандартов внедрения отечественных решений в контур безопасности предприятия, становится приоритетной задачей [2].

Подготовка специалистов в рамках дисциплины «Безопасность операционных систем» РГУ нефти и газа (НИУ) имени И. М. Губкина в 2024–25 году осуществлялась на базе учебно-исследовательской лаборатории по продуктам и решениям «Базальт СПО».

Одним из направлений подготовки было изучение подхода Infrastructure as Code (далее — IaC). IaC избавляет разработчиков и системных администраторов от необходимости вручную настраивать и обслуживать инфраструктуру. Инструмент автоматизации Ansible позволяет реализовать основные принципы IaC [3]. Ключевым элементом его работы являются модули — скрипты или программы, загружаемые и исполняемые на удалённых хостах. Модули составляют основу плейбуков — текстовых файлов, содержащих инструкции на языке YAML.

В соответствии с базой данных угроз безопасности информации Федеральной службы по техническому и экспортному контролю плейбуки подвержены ряду уязвимостей, которые могут привести к критическим нарушениям безопасности всей инфраструктуры [4]. Например, выполнение задач с привилегиями суперпользователя может привести к выполнению вредоносных действий на удалённых узлах, если плейбук доступен неавторизованным пользователям.

Одним из ключевых шагов к выявлению и устранению уязвимостей является проверка кода на синтаксическую валидность. Такая базовая проверка позволяет убедиться, что код написан правильно с точки зрения структуры YAML. Для выполнения этой задачи существуют специализированные статические анализаторы, использование которых помогает сократить количество ошибок и упростить дальнейшее сопровождение кода.

В работе был проведён статический анализ уязвимостей в конфигурации для управления полнодисковым шифрованием LUKS в ОС Альт. Анализ включал проверку валидности YAML-кода, корректности использования модулей и определения потенциальных ошибок в логике задач.

Для выполнения анализа использовались встроенная программа проверки синтаксиса Syntax Check, а также инструменты Ansible Lint и YAMLLint. В отличие от Syntax Check, который анализирует только синтаксис плейбуков, YAMLLint выполняет более глубокую проверку, включая анализ форматирования, структуры и наличие неиспользуемых элементов. YAMLLint ориентирован на проверку отдельных YAML-файлов, в то время как Ansible Lint фокусируется на всестороннем анализе ресурсов Ansible, включая разработку ролей, корректность использования модулей и соблюдение рекомендаций по написанию плейбуков.

После устранения обнаруженных уязвимостей была проведена повторная проверка кода с использованием тех же инструментов для подтверждения исправлений.

Литература

- [1] Доктрина информационной безопасности РФ (утв. Указом Президента РФ от 5 декабря 2016 г. № 646).
- [2] Уймин, А. Г. *Разработка методики тестирования системы безопасности автоматизированных систем управления технологическими процессами на основе корпоративного стандарта* / А. Г. Уймин // Автоматизация и информатизация ТЭК. — 2024. — № 5(610). — С. 59–65. — EDN VSLWIA.
- [3] Ansible community documentation — Текст: электронный. — URL: <https://docs.ansible.com/> (дата обращения: 14.01.2025).
- [4] Банк данных угроз безопасности информации — Текст: электронный. — URL: <https://bdu.fstec.ru/threat> (дата обращения: 15.01.2025).

Ася Маркина, Дмитрий Костюк, Александр Дубицкий
Брест, Брестский государственный технический университет

Практика получения и обработки окулографических данных при тестировании пользовательских интерфейсов в GNU/Linux

Аннотация

Рассматривается использование айтрекеров потребительского сегмента для тестирования графических интерфейсов в GNU/Linux. Рассмотрена возможность применения айтрекера в случаях, когда доступен SDK, и когда его нет. Обсуждаются возможности замены айтрекеров веб-камерой или отслеживанием мыши. Представлены особенности визуализации данных айтрекеров средствами Graphviz и GNU Octave, способы получения и примеры тепловых карт.

Ключевые слова: *окулография, графический интерфейс, граф фиксации, тепловая карта.*

При взаимодействии с современным программным обеспечением зрение играет роль основного, а часто и единственного канала получения информации. Поэтому отслеживание движения глаз, известное как окулография или айтрекинг, при работе с графическим приложением позволяет решать ряд важных задач usability-тестирования:

- регистрировать источники проблем пользовательского взаимодействия (видимость и заметность элементов, смещение фокуса внимания, изменения в умственной нагрузке и воздействие отвлекающих факторов);
- выявлять особенности поведения пользователя при работе с интерфейсом (визуальные стратегии поиска, шаблоны чтения и сканирования);
- получать сравнительные результаты для оценки изменений в интерфейсе и сравнения решений (проведение А/В-тестирования).

В устройстве современных айтрекеров обычно в том или ином виде используются цифровые камеры, которые снимают с высокой частотой кадров глаза пользователя, после чего программно регистрируется перемещение точки фокусировки взгляда. Айтрекеры потребительского сегмента стационарно монтируются в поле зрения, обычно внизу экрана. В айтрекер данного типа помимо нескольких камер встроена также инфракрасная подсветка: её излучение направлено в глаза пользователя и подсвечивает зрачки в инфракрасном диапазоне, создавая на изображении, получаемом камерой, аналог эффекта «красных глаз». Программное обеспечение обрабатывает изображения с камер, вычисляет угол обзора и выполняет геометрический расчёт, результатом которого является точка фиксации взгляда на экране. В случае более дорогих исследовательских моделей возможны другие варианты, включая носимое исполнение.

Профессиональные устройства отслеживания направления взгляда, как носимые, так и стационарные, часто оказываются совместимы с Linux и другими Unix-подобными системами. Это происходит в основном потому, что необходимые вычисления выполняются встроенным микроконтроллером, который передаёт данные по какому-то распространённому протоколу (например, TCP/IP) в задокументированном формате. Такой айтрекер может иметь интерфейс подключения USB, как и айтрекер потребительского сегмента, но, например, представляться не проприетарным устройством, а универсальным сетевым адаптером, «включённым» в виртуальную сеть с сервером, от-

дающим данные в ответ на запросы клиентского ПО. Однако цена таких устройств может достигать до 10 000 долларов, и как правило, оказывается неподходящей для программного обеспечения с открытым исходным кодом. Решение потребительского уровня стоимостью в сотни долларов обычно имеет несколько меньшую точность, а кроме того, перекладывает часть расчётов на драйвер, установленный на персональном компьютере, и поэтому имеет гораздо худшую совместимость. Однако требовательность современных игр приводит к тому, что диапазоны параметров профессиональных и потребительских моделей айтрекеров пересекаются, то есть по своим характеристикам актуальные геймерские модели попадают в обе категории [6].

На другом конце ценовой шкалы находится программное обеспечение для отслеживания глаз на основе веб-камеры. Например, можно упомянуть проект `WebGazer.js` [1], но он не единственный. Такие решения не могут отслеживать поворот головы (или не могут надёжно его отслеживать), и, вероятно, пропускают какой-то процент фиксаций взгляда из-за частоты кадров камеры. Но главной проблемой является то, что разрешающая способность позиционирования взгляда оказывается в два раза ниже разрешения используемой камеры. Это не было бы такой серьёзной проблемой в случае смартфона — однако смартфон сложнее использовать в качестве устройства, стационарно прикреплённого к дисплею компьютера, а разрешение встроенных камер современных ноутбуков делает потерю точности серьёзной проблемой.

Таким образом, оптимальным выбором оказываются айтрекеры потребительского сегмента, рассчитанные на применение в компьютерных играх. Ключевым игроком в этом сегменте является фирма Tobii, производящая помимо профессиональных айтрекеров также геймерскую линейку. Эти айтрекеры имеют интерфейс USB, и стоит отметить, что режим их работы (отдача координат SDK по проприетарному протоколу или работа в режиме сетевого адаптера, отдающего координаты в виде сетевых пакетов через TCP/IP) определяется не столько моделью устройства, сколько версией установленного SDK. Платный Tobii Pro SDK, ориентированный на научное применение, поддерживает и геймерскую линейку компании, но очевидно представляет наименьший интерес для СПО-проектов.

Время от времени Tobii выпускает экспериментальную версию SDK геймерских айтрекеров специально для GNU/Linux — однако затем отказывается от её дальнейшего распространения и развития. Это

происходило минимум дважды, и таким образом, для GNU/Linux есть два варианта SDK:

- заброшенная (старая) экспериментальная версия Tobii Gaze SDK for Linux, ненадолго появившаяся в 2014 году, поддерживает старые айтрекеры (например, Tobii REX) и больше не распространяется;
- заброшенная (новая) экспериментальная версия Stream Engine SDK for Linux, ненадолго появившаяся в 2017–2018 г., которая поддерживает айтрекер Tobii 4C, но с некоторыми усилиями позволяет использовать и текущую модель Tobii Eye Tracker 5.

Даже «новый» Stream Engine SDK является достаточно устаревшим продуктом. Он создавался для версий Ubuntu, существовавших до перехода на systemd, однако позже была выполнена сторонняя адаптация [2, 3] и благодаря этому он работоспособен в более современных версиях Ubuntu и в других дистрибутивах, например, Arch Linux.

Необходимо упомянуть также и работоспособную схему отслеживания взгляда Linux-пользователей на основе двух компьютеров [4]: «измеряющего», работающего под управлением MS Windows нужной версии, и собственно компьютера с Linux и ПО для тестирования. Айтрекер закрепляется на корпусе компьютера с Linux так, чтобы распознать лицо оператора, но электрическое подключение айтрекера выполняется к разъёму «измеряющего» компьютера, который выполняет калибровку и приём данных, в то время как пользователь работает с приложением под Linux.

Результатом работы айтрекера является большой массив координат, соответствующих положению взгляда в различные моменты времени. Оба SDK для Linux включают пример, в цикле выводящий поток координат на стандартный вывод. Причём, с точки зрения Tobii, сохранение в файл, в отличие от стандартного вывода, является сохранением «личных данных пользователя» и требует отдельной лицензии — чего нельзя сказать о координатах, используемых за счёт перенаправления ввода-вывода для визуализации «на лету».

Независимо от способа получения координат, распространение получили следующие методы визуализации этих данных:

- «gaze plot» или граф фиксаций отражает последовательность фиксаций взгляда. Помимо размещения узлов графа в точках фиксации, он может передавать дополнительную информацию.

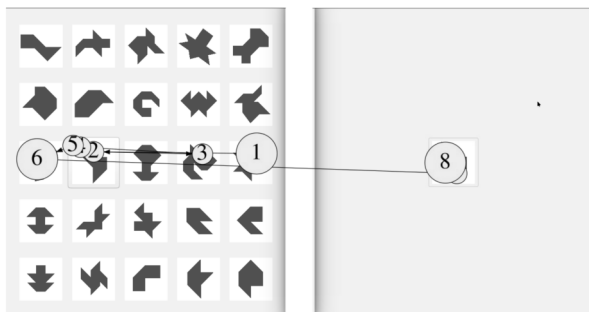


Рис. 1: Пример графа фиксаций (gaze plot)

Например, меткой узла графа может быть его номер в последовательности фиксаций, а размер узла может показывать длительность фиксации.

- «heat map» или тепловая карта показывает распределение различных цветовых температур плотности фиксаций на изображении наблюдаемого объекта (с учётом длительности фиксации или без неё). Реже встречается fog map, где размываются те части изображения, которые были просмотрены менее других.

Пример gaze plot, построенного с помощью graphviz, показан на рисунке 1. Здесь приведён тест, использовавшийся нами в ряде исследовательских проектов, измеряющий, как быстро пользователь находит определённую геометрическую фигуру из правого окна среди 25 фигур в левом окне (выполняется многократно со случайными фигурами). Как можно видеть, gaze plot подходит для статического отображения с небольшим количеством фиксаций, а большой массив координат окажется различим только при последовательном отображении нескольких gaze plot в пошаговом режиме. Преимуществом этого подхода является то, что достаточно легко создать скрипт, который преобразует отметки времени в длительность и переводит наборы координат фиксаций в формат graphviz.

Для больших статических визуализаций можно использовать Octave (без использования меток и переменного размера узлов), что позволит визуализировать массив двумя строчками кода:

```
m=csvread('data.txt');
```

```
plot(m(:,1), m(:,2), "o-");
```

Однако предпочтительным вариантом для больших объёмов данных является тепловая карта. В GNU/Octave построение теплокарты можно выполнить с помощью метода сглаживания данных, известного как двумерная «ядерная оценка плотности случайной величины» [5], а в качестве готовой к применению реализации можно посоветовать простой в использовании m-скрипт, распространяемый под лицензией BSD [7].

```
m=csvread(stdin);
x=m(:,1); y=m(:,2);
p=gkde2([x'; y']);
figure(1);
surf(p.x, p.y, p.pdf);
colormap(jet);
shading interp;
grid off; axis off;
view(2);
%saveas(1,'heatmap.png');
pause;
```

В заключение нельзя не отметить ещё один способ применения тепловой карты, который считается максимально приближённым к карте фиксаций в случаях, когда айтрекер недоступен — а именно в фиксации координат курсора мыши. Можно предположить, что в ряде задач (активно действующих курсор) будет наблюдаться некоторая корреляция между картой его координат и картой фиксаций. Существуют веб-сервисы, предлагающие такую услугу онлайн для тестирования сайтов, и действительно, результирующая тепловая карта является в определённой степени информативной, и, возможно, приближается к карте фиксаций, когда тестируются навигационные элементы, а не работа с контентом. Получение координат курсора в GNU/Linux выполняется достаточно простой командой. Например, следующее применение `xdotool` производит результат, аналогичный данным айтрекера:

```
while true; do xdotool getmouselocation | sed -e \
's/ screen:0 window:[^ ]*/g' >> data.txt; time >> data.txt;
done
```

На рисунке 2 показано сравнение карты фиксаций взгляда и карты положений курсора, полученной приведённой выше командой (в качестве задачи использовался поиск заданного товара на маркетплейсе amazon).

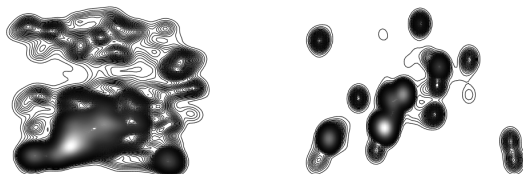


Рис. 2: Тепловая карта фиксаций взгляда (слева) и положения курсора (справа): изображения, модифицированные для черно-белой печати.

Литература

- [1] WebGazer.js: *Scalable Webcam EyeTracking Using User Interactions* / <https://github.com/brownhci/WebGazer>
- [2] Hector (Eitol) Oliveros H. *Installation and testing of tobii eye tracker in Ubuntu 18.10* / https://github.com/Eitol/tobii_eye_tracker_linux_installer
- [3] Cadu Elmadjian. *Tobii 4C for linux*. / https://github.com/johngebbie/tobii_4C_for_linux?tab=readme-ov-file
- [4] Дубицкий А. В. и др. *Применение айтрекеров для юзабилити-исследований ПО в GNU/Linux* // Четырнадцатая конференция разработчиков свободных программ: Тезисы докладов. — Калуга, 22–24 сентября 2017 г. — М.: Альт Линукс, 2017. — С. 36–41.
- [5] Tarn Duong. *Kernel density estimation for bivariate data* / <https://cran.r-project.org/web/packages/ks/vignettes/kde.pdf>
- [6] Titz J., Scholz A., Sedlmeier P. *Comparing eye trackers by correlating their eye-metric data* // Behav. Res. 2017. <https://www.ncbi.nlm.nih.gov/pubmed/28879442>.
- [7] Yi Cao. *Bivariant Kernel Density Estimation (V2.1)* — MATLAB Central File Exchange. / <https://www.mathworks.com/matlabcentral/fileexchange/19280-bivariant-kernel-density-estimation-v2-1>



Научное издание
ДВАДЦАТАЯ КОНФЕРЕНЦИЯ
СВОБОДНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ
В ВЫСШЕЙ ШКОЛЕ

Сборник материалов конференции

Переславль-Залесский
07–09 февраля 2025 года

Оформление обложки: А. А. Бусоргин.
Вёрстка: В. Л. Чёрный.
Редактура: В. Л. Чёрный.

Отпечатано с готового оригинал-макета

Подписано в печать 28.01.2025

Формат 60х90 1/16 Усл. пч. л. 9

Тираж 200 экз. Изд. 002

Издательство ООО «МАКС Пресс»

Лицензия ИД N 00510 от 01.12.99 г.

119992, ГСП-2, Москва, Ленинские горы, МГУ им. М. В. Ломоносова,

2-й учебный корпус, 527 к

Тел. 8(495)939-3890/91. Тел./Факс 8(495)939-3893

Отпечатано в полном соответствии с качеством
предоставленных материалов в ООО «Фотоэксперт»
109316, г. Москва, Волгоградский проспект, д. 42,
корп. 5, эт. 1, пом. I, ком. 6.3-23Н